APPROXIMATING COMPUTATIONAL FLUID DYNAMICS FOR GENERATIVE DESIGN

Samuel Wilkinson

February 2015

Submitted in partial fulfilment of the requirements for the degree of
Engineering Doctorate in Virtual Environments, Imaging and Visualisation.

UCL Institute for Environmental Design and Engineering

I, Samuel Wilkinson, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Samuel Wilkinson
February 2015

Abstract

Wind loads are a critical consideration in the early-stage design of tall buildings for mitigation of wind-induced forces through form modification. Existing research in computational fluid dynamics (CFD) development tends either towards fast-inaccurate or slow-accurate approaches; therefore offering either constrictive response time or inadequate accuracy. Novel approaches that combine both speed and accuracy are required to keep pace with developments in parametric design softwares, such as *GenerativeComponents*. These software tools, primarily used in early-stage generative design, allow for broad exploration and optimisation within the potential design space, which in turn requires commensurate fast-yet-accurate analysis tools. This thesis investigates the use of reduced-order models to approximate CFD simulations of wind pressure on tall buildings. It is hypothesised that: firstly, wind-induced surface pressure on tall buildings simulated by CFD can be locally approximated by geometric features; and secondly, reduced-order model predictions dominate CFD simulations in both time or accuracy and are therefore a novel non-dominated approach. Predictions are made of individual vertex pressure based on input features formed from local shape analysis. The vertex samples originate from a procedural model set which is evaluated with either steady-state Reynolds-averaged Navier-Stokes (RANS) or transient large eddy simulation (LES). An artificial neural network is used for model reduction with the training set of vertex samples; the basis methodology of which is tested on a range of study complexities. To prove the scalability of the approach, this culminates in the use of LES as the basis simulation, a test set of realistically complex building models, and an alternative approach to urban wind interference generalisation is also described, whereby a one-off large-scale context CFD simulation can be used as input to repeatable design model predictions. Furthermore, a prototype tool and an outline for its integration with an existing online analysis framework currently under development is presented. The quantitative and qualitative results of the studies show it is possible to approximate surface pressure from local shape features, thereby decoupling the prediction from the basis simulation. The reduced-order model can achieve fast-yet-accurate results, since prediction accuracy and time are invariant, or independent, of basis simulation accuracy and time; being instead solely a function of the reduced-order model performance and the geometric complexity or number of test mesh vertices. Evidence is demonstrated by the positioning of the results as a non-dominated solution in the time-accuracy objective space and the subsequent alteration of the existing Pareto frontier.

Publications

The following were published or submitted in relation to material contained in this thesis. All publications are included for reference in Appendix A.

Journal Publications

Wilkinson, S., Bradbury, G., and Hanna, S. (2015). Reduced-Order Urban Wind Interference. *Journal of Simulation.*

Wilkinson, S. and Hanna, S. (2014). Approximating Computational Fluid Dynamics for Generative Tall Building Design. *International Journal of Architectural Computing*, 12(2), pp.155-178.

Conference Proceedings

Wilkinson, S. and Hanna, S. (2014). Reduced-Order Topological Performance Models. In *proceedings of IBPSA-CIBSE BSO14: Building Simulation and Optimisation.*

Wilkinson, S., Bradbury, G., and Hanna, S. (2014). Approximating Urban Wind Interference. *Simulation Series - Proceedings of SimAUD '14*, 46(7) pp.82-89.

Wilkinson, S., Hanna, S., Hesselgren, L., and Mueller, V. (2013). Inductive Aerodynamics. In *Computation and Performance - Proceedings of the 31st eCAADe Conference*, vol.2, pp.39-48.

Acknowledgements

This research was supported by the UK Engineering and Physical Sciences Research Council (EPSRC), in partnership with Bentley Systems and PLP Architecture. Bentley and PLP together provided the initial opportunity and ongoing motivation for the project.

I would firstly like to thank my sponsor supervisors, Lars Hesselgren and Volker Mueller, for their enthusiasm and the endless supply of ideas over the past four years. I am grateful for Alasdair Turner welcoming me kindly to the Bartlett; and to my secondary supervisor Tadj Oreszczyn for ensuring my progress since then. My greatest thanks go to Sean Hanna for guiding me through my time at UCL with constant clarity, wisdom, and encouragement.

In the end, I am most grateful for the unbounded discussions with colleagues and friends, and the continuing reassurance of my family, without which this would not have been possible.

London, February 2015
Samuel Wilkinson

Acronyms

AEC	Architecture, Engineering, and Construction
ANN	Artificial Neural Network
API	Application Programming Interface
BLWT	Boundary-Layer Wind-Tunnel
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
CFX	*ANSYS CFX*
CPU	Central Processing Unit
CTBUH	Council on Tall Buildings and Urban Habitat
CWE	Computational Wind Engineering
DAG	Directed Acyclic Graph
DNS	Direct Numerical Simulation
DVM	Discrete Vortex Method
FEM	Finite Element Method
FFD	Fast Fluid Dynamics
GA	Genetic Algorithm
GC	*Bentley GenerativeComponents*
GPU	Graphics Processing Unit
IF	Interference Factor
k-ϵ	k-Epsilon turbulence model
KDE	Kernel Density Estimation
LBM	Lattice-Boltzmann Method
LES	Large Eddy Simulation
LS-SVM	Least-Squares Support Vector Machine
OM	Obstruction Model
PDE	Partial Differential Equation
PM	Principal Model
RANS	Reynolds-Averaged Navier-Stokes
RBF	Radial Basis Function
RF	Random Forest
RIBA	*Royal Institute of British Architects*
RMS	Root-Mean-Square
RNG	Re-Normalisation Group
ROM	Reduced-Order Model
SAT	Speed-Accuracy Tradeoff
SOM	Self-Organising Map
SPH	Smoothed-Particle Hydrodynamics
SVM	Support Vector Machine
WALE	Wall-Adapting Local Eddy viscosity model
WT	Wind-Tunnel

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Definition

The accelerating ability of computers to simulate, prior to realisation, the complex behaviours of buildings is further extending design potential, performance, and our understanding. Affordances are therefore being created to consider a broader range of options with a finer degree of inquisition; a trend being led by hardware and software developments, and closely shadowed by parametric design tools. Recently, there has been an active interest from the architectural community in parametric CAD and simulation, with a subsequent motivation to integrate analysis and geometry towards a semi-automated generative exploration and optimisation.

Some of the most complex simulations to be used for building performance analysis are those involving partial differential equations (PDEs) in their approximation of the underlying physics. Within this category the most complex again is turbulent wind flow. Such fluid simulation becomes particularly significant for tall buildings, bridges and urban design, where the wind plays a dominant role in structural behaviour and occupant comfort. The focus of this thesis is on tall buildings as there has always been a drive to build as high as possible, to the supertall and megatall generation of today where aerodynamics is an intrinsic part of their design.

The quantity and height of tall buildings is increasing, primarily within the constraints of the economy and technology. As height increases the wind becomes a dominant force, and the overall architectural building form becomes a significant factor in its behaviour. Such large scale form, or massing, decisions are typically made at early design stages, and so even though they may have the greatest impact they occur with the least guidance or analysis. Notionally, mitigating wind loads can consequentially improve the structural efficiency, reduce construction costs and material, increase revenue, and allow for even taller construction.

There are two approaches to mitigation: the first is through aerodynamic modifications typically occurring at later design stages after the primary form has been established, and generally involves

corner modifications such as chamfering or cutting; the second, and most effective, is by aerodynamic design of the global form of the structure which must ideally take place from the start of a project. The distinction must also be drawn between along-wind and across-wind structural responses: along-wind being the steady-state drag of the building; and across-wind the periodic forces on the building arising from vortex excitation and synchronisation (J. Xie, 2014). The focal application of this thesis is on generative aerodynamic design for along-wind responses.

The inherent problem therefore becomes apparent: incorporating aerodynamic performance effectively in early design requires a minimal response time, yet achieving reasonably accurate results involves more time than is practical. Wind simulation time is currently unsupportive of the fast, iterative nature of parametric design and optimisation. The development of computational fluid dynamic (CFD) solvers has generally been within the fields of either aerospace engineering or computer graphics. At different extremities of a speed-accuracy tradeoff (SAT) (Chittka et al., 2009), the first places accuracy before speed, and vice versa for the latter. Since architects are increasingly wanting access to this analysis data their choice is therefore between slow-and-accurate, or fast-and-inaccurate.

Wind engineering for tall buildings has, for the past fifty years, been almost exclusively within the remit of specialist engineers and wind-tunnel technicians. The success of boundary layer wind-tunnel (BLWT) physical experiments has on one hand created considerable scrutiny (Bitsuamlak, 2006; Dagnew et al., 2009; Menicovich et al., 2002; Stathopoulos, 1997) of the accuracy of a purely numerical approach (CFD), and yet it has also acted as a driving force for improvement (Blocken (2014) gives a thorough history of developments in computational wind engineering (CWE) over the past 50 years). Whilst BLWT is still demanded for most projects at latter stages, as CFD has improved it has taken an increasingly supportive role especially at the middle stages of the design prior to BLWT. Here it has shown its advantage of comparative speed, knowledge output and, arguably, cost. However, at early design stages where the interest is on relative comparison and general flow behaviour (Lomax et al., 2001; Malkawi et al., 2005), CFD is still not able to offer a solution that is both accurate and fast.

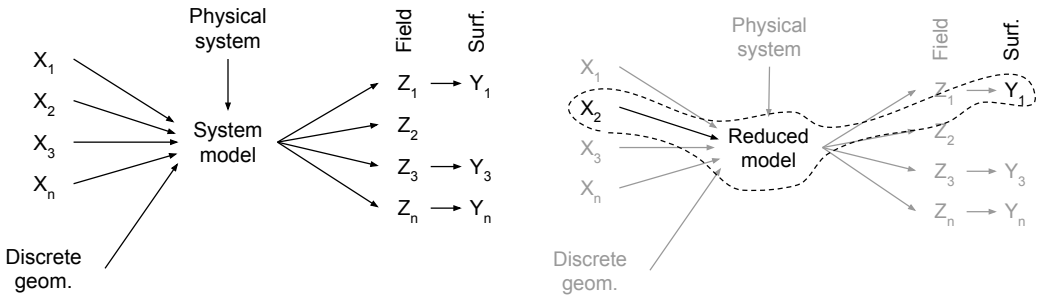
The proposed solution that is explored here to achieve speed and accuracy is through the use of machine learning. Learning by artificial neural networks (ANN), support vector machines (SVM), and random forest (RF) decision trees all use a training set of cases from which generalised rules are generated (Duffy, 1997). The outcome of this approach is an approximate model of the simulation output, where the simulation time is transferred from the front- to back-end of the process when more time is available for pre-computation of the training set.

The novel approach of this thesis is the local definition of the learning problem, as opposed to what might be considered the obvious top-down definition. In creating a relationship between a building's

geometry and its aerodynamic behaviour, the first step is to create a description of the geometry with which to relate the simulation output. The initial obvious description is by a top-down, or global, descriptor: for instance, a skyscraper can be described by its height, planform shape, or degree of taper. But these global input descriptors can only lead to a global performance output, as demonstrated in §5.2. In contrast to the top-down approach, examples are given where local geometry (vertex) descriptors are related to a local output (vertex wind pressure). It is shown how this localisation of the relationship increases the applicability of the method considerably.

1.2 Approximation

Approximation is a key concept to this work and in simulation methodology in general. The distinction is introduced here between solver and solution approximations. Solver approximation refers to the a priori reduction involved in the simulation of the underlying physical system, i.e. the various approaches of CFD simulation (Figure 1.1a); with boundary conditions (X) and geometry as input; and direct field (Z) and derived surface (Y) simulation outputs. Solution approximation refers to the a posteriori approximation of the CFD simulation, i.e. generation of the reduced-order model or machine learning typically between input boundary conditions and derived outputs (Figure 1.1b shows the typical configuration between boundary conditions and derived output, e.g. for an airfoil, X=Mach speed $\rightarrow$ Y=lift). The difference between this conventional configuration and the proposed approach is detailed in Chapter 5. The terms *simulation* (Y) and *prediction* (Y') are used in subsequent chapters to distinguish between the two.



(a) Solver approximation (b) Solution approximation

Figure 1.1: Approximation approach comparison.

Whereas the simulation accuracy is measured against reality (or what is referred to as 'experimental' data, although this terminology may be dated with the increased acceptance of simulation as a type of experiment), predictions are measured relative to the CFD simulation of which it is approximating. The objective of the first is fundamentally to achieve verisimilitude (*'degree of truth'* or the *'truth content of a theory and of its approximation or nearness to truth'* as described by Rohrlich (1990) and Popper (1959)) and therefore the emphasis is on accuracy over speed. Whilst

CFD originating in engineering disciplines aims for physical accuracy, fast fluid dynamics (FFD) and other computer graphics fluid simulations aim only for visual accuracy. Indeed, establishing the accuracy of CFD against a ground-truth is a non-trivial process and will be discussed in greater depth in §4.1. In contrast, prediction accuracy is measured relative to the CFD simulation it is approximating and is therefore relatively straightforward by comparison (§5.1). The two methods for calculating the prediction accuracy, sample- and model-based, are described in Chapter 5.

Within existing CFD simulation approaches there is a general tradeoff between accuracy and speed, ranging from fast-inaccurate to slow-accurate (§3.1.2). Since the approach objectives of speed and accuracy are essentially opposed within traditional solver approximations, the tradeoff can be expressed as a multi-objective problem. Within this framework, each approach or solution can be positioned in a time-accuracy objective space; the solutions fitting a Pareto frontier are optimal to some extent with regards to either objective. The solutions forming the Pareto frontier dominate the rest, creating the distinction between *dominated* and *non-dominated* sets: one solution dominates another if both objectives are at least as good as other solutions; and at least one objective is better. The second hypothesis defined below rests on the positioning of existing (solver approximation) and proposed (solution approximation) methods, and measurement of any alteration to the Pareto frontier.

1.3 Hypotheses

There are two hypotheses put forward within this thesis: the first relates generally to the theoretical premise of the approach; and the second to the implications of its validity.

i) Surface pressure simulated by CFD can be approximated by local geometric features.

Within the whole fluid domain, the nonlinearity and continuity of fluids causes a spatial and temporal dependence or sensitivity between all elements, which suggests that full domain simulation is necessary. The first hypothesis seeks to prove that through model reduction the simulated pressure can be approximated or learnt by local shape features. This implies that aspects of the simulated fluid domain, namely between the geometric description and pressure of a vertex, can be reduced to tractable relationships suitable for learning.

ii) Reduced-order model predictions dominate CFD simulations in both time and accuracy.

Existing fluid simulation methods tend towards being either fast-inaccurate or slow-accurate, whereas the reduced-order model can achieve fast-yet-accurate results. This is possible since the prediction accuracy and time of the reduced-order model are invariant, or independent, of basis simulation accuracy and time. Evidence is demonstrated by the positioning of the results as a non-dominated solution in the time-accuracy objective space and the subsequent alteration of the

existing Pareto frontier.

1.4 Outline

The thesis can be broadly divided into three parts: Context (Chapters 1, 2 and 3); Experimentation (Chapters 4, 5 and 6); and Analysis (Chapters 7 and 8). In the second part, the methodology and results are largely presented together since there are a number of separate experimental studies. The thesis narrative structured by chapter is introduced below:

Chapter 2 - Applied Research: The basic motivation and application for this research is established here in the background context of tall building design. The two research partners, *Bentley* and *PLP*, contribute by providing real problems, constraints, and development opportunities. In Chapter 2, the symbiosis between software developer and user is considered by analysing the mutual benefits of tool development. The current development of *Bentley's* online parametric analysis framework and *PLP's* requirements for analysis in generative tall building design set the commercial context for the research. This context sets the framework proposed in Chapter 7 of integrating an online reduced-order model and feature database.

Chapter 3 - Literature Review: The current state of the art in relevant fields is necessarily divided into three areas for review: i) contemporary architectural practice, particularly performance-guided generative design; ii) computational fluid dynamic simulation of the wind around tall buildings; and iii) solution approximation with machine learning for design applications. As well as establishing current trends or trajectories, deficiencies are also identified; these can be summarised as movement towards more complex tall building forms being designed with generative CAD tools and a lack of CFD analysis tools tailored to the specific speed and accuracy requirements. A key contribution of this chapter is identifying existing sources of CFD validation data against which the later experimental studies can be positioned.

Chapter 4 - Fluid Simulation: The speed and accuracy of existing fluid simulation approaches are assessed in the fourth chapter through comparative validation against source data from the literature. The first approach, RANS CFD, is ubiquitous in engineering where it is generally accepted for its accuracy (LES is still not widely used in practice and DNS remains purely for research (Blocken, 2014; Stathopoulos, 1997)), and is the basis of subsequent approximations. The second approach, fast fluid dynamics (FFD), is being applied in architectural practice primarily because of its speed, and therefore gives a benchmark against which to compare a new method for the same application and establish an 'acceptable' degree of accuracy and speed. In order to position these in a broader framework, higher-order LES and wind-tunnel data are also assessed. Through this validation, the ranges of existing speed and accuracy are defined, allowing for relative positioning of the proposed approach.

Chapter 5 - Approximations: The basic methodology of the approximation approach is introduced in the fifth chapter, where a number of studies of increasing complexity are described along with an analysis of their results. A global approximation method is initially presented in order to highlight certain limitations that instigate the local approximation method, followed by development of the local shape feature definition. The novel approach of this thesis is this local definition of the learning problem, as opposed to what might be considered the obvious top-down definition. It is shown how this localisation of the relationship increases the applicability of the method considerably. All of the studies are generally trivial in a design context but provide the foundation methodology and results for supporting the hypotheses. In this respect, the prediction time and accuracy of each study are assessed relative to the basis CFD of which they are approximating. Sensitivity studies are also conducted for feature selection, learning parameters, and generalisation comparison of the random forest and artificial neural network algorithms.

Chapter 6 - Complex Applications: In the sixth chapter the methodology is extended as a test of the scalability to realistically complex scenarios that would be found in practice. Three aspects of realistic design applications are addressed by extending the basic methodology set out previously: i) using peak pressure from transient flows with unsteady behaviour using LES as the basis for approximation, instead of the previous time-averaged RANS basis. This tests the applicability of the reduced-order model (ROM) to another basis CFD that is slower and more accurate than RANS, and to temporally distributed pressure variation; ii) complex geometry of real tall building tests from a training set of procedural models. This is a direct evolution and step up in complexity from all of the local shape studies in Chapter 5; and iii) wind interference in dense urban contexts with the addition of local wind speed as a feature. This scales up the final tests in Chapter 5 to a far greater complexity obstruction model, and includes further sensitivity analyses on the feature selection and training set configuration. Results from both Chapters 5 and 6 provide the data for compilation in Chapter 7 and for a direct response to the hypotheses in Chapter 8.

Chapter 7 - Discussion: In this chapter the experimental results of the previous two chapters are discussed in relation to the validation results from Chapter 4. This allows for the positioning of the solver (FFD and CFD) and solution (Reduced-Order Model) approximation results in the time-accuracy domain. Analysis of the existing and subsequent approaches allows a response to the second hypothesis as to the positioning of the reduced-order model relative to CFD. This is followed by a discussion on the potential for an open-source tool in line with ongoing developments and a summary of the methodological constraints.

Chapter 8 - Conclusion: Finally, responses to the hypotheses confirming the validity of both, a summary of the key findings, contributions, and recommendations for further work are given.

Appendices: Publication are given in Appendix A and code in Appendix B.

Chapter 2

Applied Research

2.1 Tall Buildings

Tall buildings are simultaneously a simple and complex building typology: simple in their basic objective of height, yet complex for the challenges this creates. With these challenges in mind, a selection of which are identified here, subsequent sections will seek to define the problems and propose solutions. In this section, the focal typology is introduced and justified for study.

In a comprehensive peer-review of the state of research in tall buildings by Oldfield et al. (2014), a questionnaire was undertaken in 2012 of practitioners and academics identifying key areas of priority and immaturity in the research field. A preliminary questionnaire was used to identify 358 topics over 11 fields, followed up with a secondary questionnaire with 347 participants. Of these, 62 responded to the field of 'Structural Performance, Multi-Hazard Design and Geotechnics'. Two topics are identified here as most relevant; each is given an Importance (1 = not at all important, 5 = extremely important) and an Immaturity rating (1 = not at all immature, 5 = extremely immature).

Firstly, *'Development of approximate tools for optimization in the early stages of high-rise design for wind (including aerodynamic databases and other approximate tools and rules based on shape, height, slenderness, exposure, structural system, etc.)'* received an Importance rating of 4.0, and an Immaturity rating of 3.4. And secondly, *'Research on the use of computational fluid dynamic tools and models in the structural/wind design of tall buildings'* received an Importance rating of 3.7 and an Immaturity rating of 3.6. The relatively high Immaturity and Importance ratings of both topics supports the timeliness of this research.

2.1.1 Construction trends

The number of tall buildings constructed globally, and their height, appears to be increasing exponentially (Figure 2.1). In this figure, black points show the existing number over $150m$ =

3004; and grey the number under construction or topped-out over $150m = 828$. These rates are affected by a number of factors, such as: financial investment; planning regulations or politics; and engineering constraints (materials, structural systems, vertical transportation, seismic or wind loads, etc.). The tallest, cutting-edge buildings are likely to be pushing the boundaries in one or more of these categories (or even going too far, i.e. taking a financial loss, structural failure, etc.). This characteristic to reach new heights is one reason for the continuing interest and developments in skyscrapers.

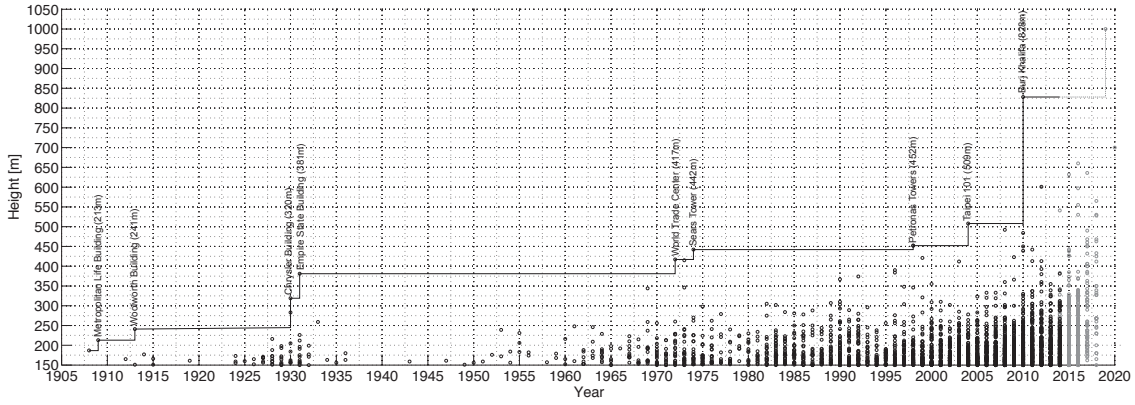


Figure 2.1: Number of tall buildings over $150m$ from 1905 to 2020, data source: CTBUH (2014).

The changing height and shape of the world's tallest 20 buildings in 2000, 2010, and 2020 are shown in Figure 2.2. Note that none of those in the 2020 set are in the 2000 set, indicating a completely new tallest population after 20 years. According to Tanaka et al. (2012), 56% of the tallest 100 buildings were completed since 2000. And in buildings over $200m$, 265 were constructed prior to the year 2000 and double this, 518, were completed between 2000 and the end of 2012 (Oldfield et al., 2014). At this time the Kingdom Tower in Jeddah is currently starting construction, and if completed will overtake the current world's tallest (Burj Khalifa, Dubai, at $829.8m$) to be the first building over $1000m$ tall.

(a) 2000 (Average: $375m$) (b) 2010 (Average: $439m$) (c) 2020 (Average: $598m$)

Figure 2.2: Skylines of the tallest 20 buildings in: a) 2000; b) 2010; and c) 2020 (CTBUH, 2011).

Despite, or because of, the rapid advances over the past century, this new generation poses serious challenges for the academic and practicing wind engineering community. Irwin (2009) identifies three areas where understanding or practice needs to be strengthened. Along with developing more

advanced vertical wind profiles for use above $300m$ (for 'supertall' buildings[1]) and models of the complex tradeoffs that aerodynamics have on cost, it is suggested that the significance of shape aerodynamics needs to be proactively considered and iteratively optimised at early stages. Using the Burj Khalifa as an example, it is described how a large amount of time was spent refining the aerodynamic behaviour of the design. By doing this it was possible to produce a more efficient shape that enabled the great height with reasonable structural systems and costs, and without any dampening system.

2.1.2 Wind effects

As height increases so too do the wind forces impacting on the building, so that out of all typologies it is perhaps the most subjected to the wind. These forces can cause a series of negative consequences that may be mitigated through form-finding at design stage. A number of these are outlined by Y. Tamura (2009) in a comprehensive review of the field:

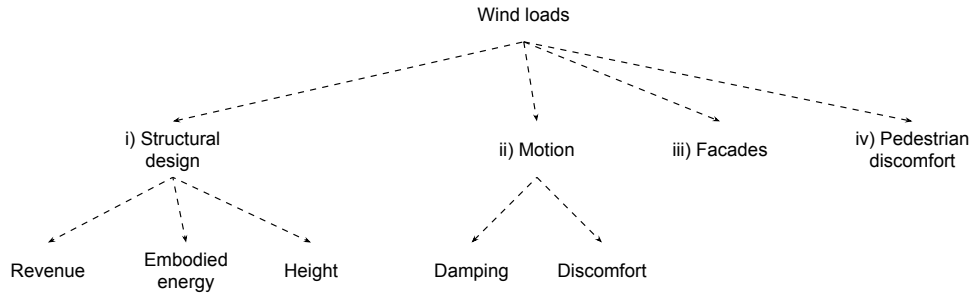


Figure 2.3: Primary and secondary effects of wind loads on tall buildings.

i) *Structural Design:* Structural inefficiencies can be caused by having to increase the amount of structure (larger cores, columns, or beams) to counter wind loads that could otherwise be mitigated by manipulation of shape, geometry, surfaces and mass distribution of the form (Watts et al., 2007). This has a number of subsequent effects:

Revenue: The structural efficiency is a measure of the amount of support material against usable floor space, i.e. the net:gross floor area ratio is the proportion of tenant space to total area constructed (CTBUH, 2010). In the cases of buildings, let-able floor area is obviously a source of revenue and can govern the financial success or investment return period. Larger cores, columns, or other structural systems interfere with this by either reducing the amount of let-able floor area or reducing its quality. The purchasing of materials itself is also an investment and is greater for less efficient structures.

Embodied Energy: The amount of structural materials used (namely concrete and steel)

[1]In current terminology, 'supertall' is $>300m$ and 'megatall' $>600m$ (CTBUH, 2013). A future-proof naming convention can be based on the orders of metric prefixes, i.e. 'kilotall' for $>300m$, 'megatall' for $>600m$, 'gigatall' for $>900m$, 'tera-', 'peta-', 'exa-', 'zetta-', through to 'yottatall' for $>2.4km$.

has an impact on the embodied energy or carbon of a building. Tall buildings generally require a greater investment of initial embodied energy per unit gross floor area in comparison to low-rise buildings. Comparing embodied energy to operational energy, for a typical good practice UK office building[2], the embodied to operational ratio is $8.1GJ/m^2 : 2.3GJ/m^2/year$ (CTBUH, 2009).

Height: Less efficient structural systems can reduce the potential or design height of a building by increased construction costs or impracticality. In practice the total height of a tall building may not even be known when construction starts, due to lengthy construction times, fluctuations in material costs, and other financial considerations. There is therefore a complex relationship between the structural design, the quantity of construction material, the investment cost, and the final height. A more efficient structure will reduce the amount of material, the construction costs, and subdue implications of material costs by supply market fluctuations.

ii) *Motion:* Under certain wind conditions and for certain building shapes, across-wind loads can cause oscillatory motion arising from vortex shedding. This can either be avoided a priori by disrupting the coherence of vortices with a changing sectional shape with height or corner modifications; or mitigated a posteriori through dampening.

Damping: Motion can be mitigated by increased structural reinforcement/stiffening (Structural Design) to change the resonant frequency of the structure; or through the installation of costly active or passive damping devices. Such devices might include: passive tuned-mass dampers; active mass dampers; sloshing-type tuned-liquid dampers; and viscous fluid dampers. Damping systems are preferred by design teams since it frees them from the need of mitigation by structural reinforcement or avoidance through form modifications or 'constraints' (Irwin, 2009). This argument, towards creating 'un-aerodynamic' buildings and fixing them with damping, is suggestive of a discrepancy between architecture and engineering or a lack of available analysis tools in the design process.

Discomfort: Wind excitation, particularly in the new generation of taller, lighter buildings, can cause low frequency, low acceleration building motion typically between 0.08 and $1Hz$ (Lamb et al., 2013). These vibrations are perceptible to occupants, potentially causing fear, headaches, stress, and motion-sickness (nausea), in turn affecting work performance and comfort.

iii) *Facades:* Under extreme conditions it is possible for facades to be damaged by high wind speeds, specifically for glazed panels where deformations can cause frame dislocation.

iv) *Pedestrian Discomfort:* The addition of any new building into an urban setting is likely to

[2]Embodied energy, $8.1GJ/m^2 = 5.2$ {47.2% recycled steel} + 1.8 {concrete} + 1.1 {33% recycled aluminium}

affect the local wind environment. Changes to wind patterns or speeds may alter the suitability of ground-level spaces for certain functions or in extreme cases make them dangerous.

As mentioned, there are two approaches to mitigation: either by aerodynamic modifications at later design stages after the primary form has been established, e.g. by corner modifications such as chamfering or cutting; or, most effectively, by aerodynamic design of the global form of the structure at early project stage. The focal application of this thesis is on such early-stage aerodynamic design.

The structural response to dynamic wind loads is not explicitly calculated in this thesis as it is considered outside of the remit of early-stage design. The Strouhal number is typically used to describe the oscillating behaviour in structures arising from cross-loads and vortex shedding (Davenport, 1995). It is given as:

$$St = \frac{f.L}{U} \tag{2.1}$$

where f is the frequency of vortex shedding, L is the characteristic length (building width), and U is the wind speed. Although the Strouhal number is a key parameter to calculate when assessing the aerodynamic behaviour of tall buildings, it is a global metric and is therefore less applicable to the scope of this thesis.

2.1.3 Complex form

In recent years there has been an increase in shape complexity beyond the traditional extruded rectilinear form. Unconventional free-style forms are now derived from the architect's use of more advanced modelling software (Y. Tamura, 2009; Y. Tamura et al., 2010; Tanaka et al., 2012), and from digital fabrication and coordination techniques. This can be explained by the integration of computation into the design and construction process at nearly every stage. Improvements in CAD and analysis have meant it is now easier to create and assess these complex forms.

Irregular forms generally increase the complexity of the building's performance or behaviour, or at least its analysis. One consequence is that with vertical sectional variation, vortices that form behind the building lose their coherence and therefore their strength. This reduces the across-wind loads and motion response of the building. In common parlance, when this strategy is used it is said to 'confuse the wind', for example in reference to the Imperial Tower, Mumbai (Campbell-Dollaghan, 2013), and the Burj Khalifa, Dubai (W. F. Baker & Pawlikowski, 2011).

A series of wind-tunnel tests were conducted on 31 tall building models (Figure 2.4) with various 'unconventional' configurations (Y. Tamura et al., 2010; Tanaka et al., 2012). This is perhaps the broadest shape sensitivity analysis conducted in a wind-tunnel to date.

It was found that the 4-Tapered and Setback models had good behaviour in the along-wind direc-

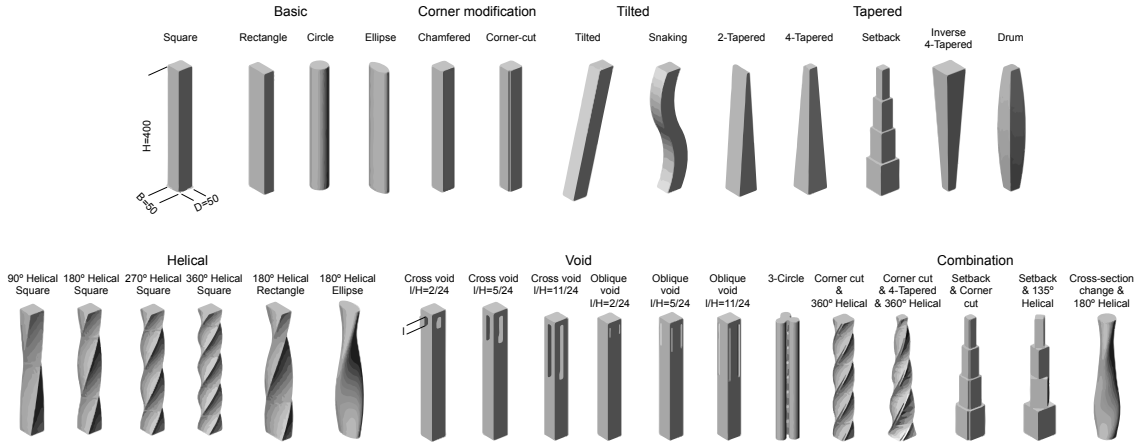


Figure 2.4: Range of unconventional models in shape sensitivity analysis (Y. Tamura et al., 2010).

tion, whilst the Corner Modification, Helical, and Cross Void models were best in the across-wind direction. Although it is a broad systematic form exploration, the complexity of the models is still relatively simplistic in comparison with current real buildings. This is a general problem with top-down design space exploration using generative or procedural models; discussed further in the following section.

Figure 2.5 shows the tallest building constructed each year between 2000 and 2012 (CTBUH, 2012). A prevalent architectural style is difficult to identify in this set, although each can be characterised by salient iconic features; such as voids, stepping, bundling, and tapering. It is therefore apparent that any top-down or global generalisation about performance or wind behaviour in the set presents a challenge.

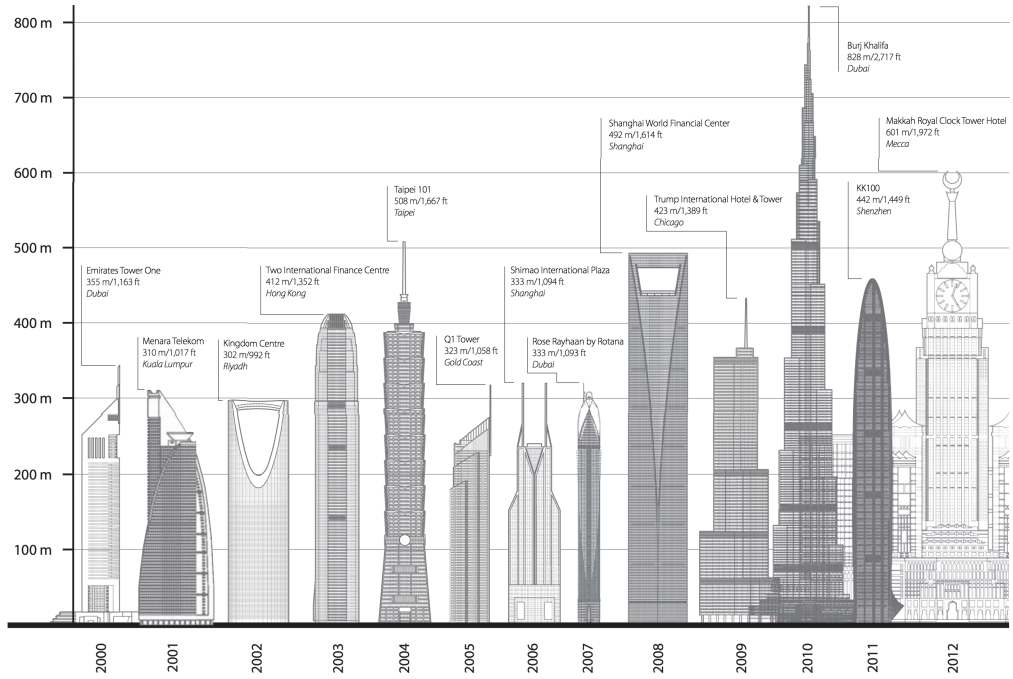


Figure 2.5: Tallest building completed each year between 2000 and 2012 (CTBUH, 2012).

2.1.4 Procedural models

Procedural, or generative, models are constructed from rules or relationships rather than explicit dimensions. For example, a procedural cube would have parameters *width*, *depth* and *height*, each with a specified parameter range. By randomising or incrementing the parameter values over their ranges, the parameter space is sampled at a given resolution. The resulting set of models represents a design space defined by the parameters and their ranges.

In the case of tall buildings, it is also possible to create a procedural model from which a large number of unique instances can be generated. The case is more complex than a simple platonic object as the parameters are less obvious and dependent on the design concept. A tradeoff arises between the number of parameters (i.e. the simplicity of the model) and the complexity or size of the resulting design space. Notionally similar to an eigenvector in principle component analysis, the addition of more parameters (dimensions) allows for more detail and complexity. An optimal model (or eigenvector) is one that has minimal input and maximum output.

There are limits however to the scope of a single procedural model, and often a new design topology requires a new model (Samareh, 2001). This is a common problem of parametric software in practice: the forethought and time invested to construct a parametric model only allows localised flexibility in the design space. It is therefore difficult to generalise between projects to have a generic set of design parameters, since each project has its own nuances and constraints.

Park et al. (2004) developed a procedural tall building model, with nine parameters such as: height; size and shape of footprint (Figure 2.6 left); and vertical transformations such as section morph, setback, twist or curvilinear profile (Figure 2.6 right).

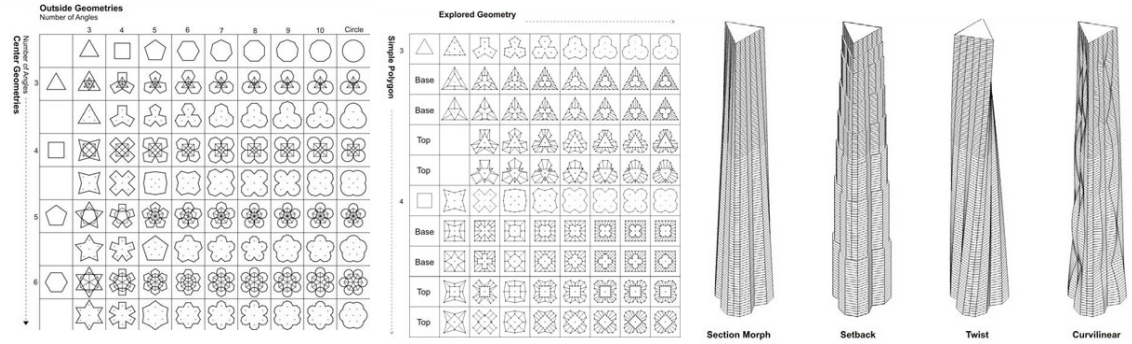


Figure 2.6: Examples of procedural tall building models (Park et al., 2004).

Procedural building models are also commonly used in various computer graphics fields, such as for video games and films (Watson et al., 2008). The objective in these cases is to automatically generate urban environments at the city-scale, which can even happen on-demand during game-play. One of the challenges, for example in CityEngine (Esri, 2014), however is to include enough parameters or randomness to capture the variability, or 'iconicity' that characterises skyscrapers.

2.1.5 Tall building typology

The reasons outlined in this first section describe the suitability and challenges of tall buildings as the focal typology of the thesis; summarised as:

- *Quantity and Height:* The increasing number and height of tall buildings around the world makes research on the typology more critical. Greater quantity simply results in a greater number of people occupying this type of building; and greater height exacerbates existing research, design, and construction challenges (§2.1.1).
- *Aerodynamics:* More so than for other typologies, tall buildings are exposed to and therefore subject to the effects of wind flow (§2.1.2). Wind engineering integrated with design is therefore an area of active research, primarily in the improvement of analysis and process tools.
- *Form Significance:* The aerodynamic behaviour is largely dependent on the overall form of the structure (along with flow conditions and surface geometry), which is typically decided upon at early project stages (§2.1.3).
- *Parametrisation:* Tall building form lends itself well to parametric design as there is often a strong vertical design logic that can be expressed mathematical and efficiently. The typology is therefore more amenable to generative and procedural modelling (§2.1.4).

2.2 Development Partnership

In the rest of this chapter the commercial context for the research will be defined, since the existing challenges and future opportunities of the partners help to position the work with realistic constraints. The two commercial partners are *Bentley Systems* and *PLP Architecture*, who as software developer and architect fit in the general classes of tool-maker and tool-user. This general relationship has a historical precedent and is characterised by a reciprocity or symbiosis focused on the mutual development of tools. Evolution of a tool can be driven by either new capabilities from the developer or by demand from the user; in either case, the tool represents a solution to a problem that has been predicted or encountered.

A key component of the reciprocal relationship is feedback, either through the sharing of ideas, problems, or solutions. The initial brief was to analyse the user's design processes to identify and generalise problems being faced in practice, then to prototype and test potential tools to solve them. From this perspective, *PLP* supply the problem constraints and *Bentley* the opportunities for development.

2.3 Tool-Maker

Bentley are a CAD software developer for the architecture, engineering, and construction (AEC) sector, with *MicroStation* for 2D or 3D modelling and the parametric modelling extension *GenerativeComponents* (*GC*) (Bentley Systems, 2013). There are also a number of analysis packages and interfaces in their portfolio, such as *STAAD.Pro* (structural analysis), *EnergyPlus* (dynamic energy analysis), and the analytic framework (a multi-objective genetic algorithm). These have recently been through beta-testing as cloud-based services integrated with *GC*, which is the intended platform for this research.

2.3.1 Parametric software

GC is a parametric CAD tool built on the principle of a dependency graph (i.e. a directed acyclic graph or propagation network). This enables associativity between model elements and variables from which complex relational dependencies can be constructed. Whilst the advantages of parametric tools are relatively well-known, as advertised, the true limitations emerge with use in real design practice (Aish & Woodbury (2005) and Woodbury (2010) give a definitive explanation and discussion of the benefits and drawbacks of parametric design tools). For instance, limited representational flexibility with complex models can hinder effective design exploration (Gürsel Dino, 2012). That is, it is effective to have a parametric model when it is initially known what the important parameters are and how to construct the logic of the model, but it is typically necessary to recreate the model if this logic changes.

The significant functionality of *GC* is explained here to understand the potential role that analysis has in its future:

- *Scripting:* There are a variety of ways to script in *GC*, such as via single-line expressions, transaction scripts, function methods or through the C# API. Each allows for mathematical functions to deal with complexity or repetition, i.e. *for* loops, logic statements, matrix (arrays, collections, or lists) operations, etc. Scripting is typically the user entry point for customising existing tools or creating their own;
- *Replication:* The Cartesian product of input collections or arrays, i.e. the Cartesian product A x B of collections A and B is the collection of all possible ordered pairs. A classic example is a deck of cards: the two individual lists are {ace, 2, 3 ... jack, queen, king} and {club, diamond, heart, spade}, and the full deck is every combination of both lists. The opportunities of being able to manipulate this becomes clear early on with the ease of populating complex surfaces with replicated elements;
- *Deferral:* Elements can be given temporary definitions and later altered so that dimensions

can be changed at any stage or defined as properties of other elements. Deferral is even more useful in the latest version of the software where it is possible to give incomplete as well as temporary definitions;

- *Dependency:* At the core of parametric software is the dependency graph, or the associative relationship between dependent and independent elements. This allows for persistent definitions such as for a point to be persistently placed at the mid-point of a curve or the number of steps to be a function of the floor-to-floor height. Dependency is not limited to geometry, instead an element's definition can be informed by any suitably formatted data. Analysis-based dependencies can require time for evaluation and to return a response. This can range from instantaneous (quantity, area, volume, or cost) to minutes (energy or solar) to hours (structural and CFD). Obviously the dependency becomes strained as the response time increases.

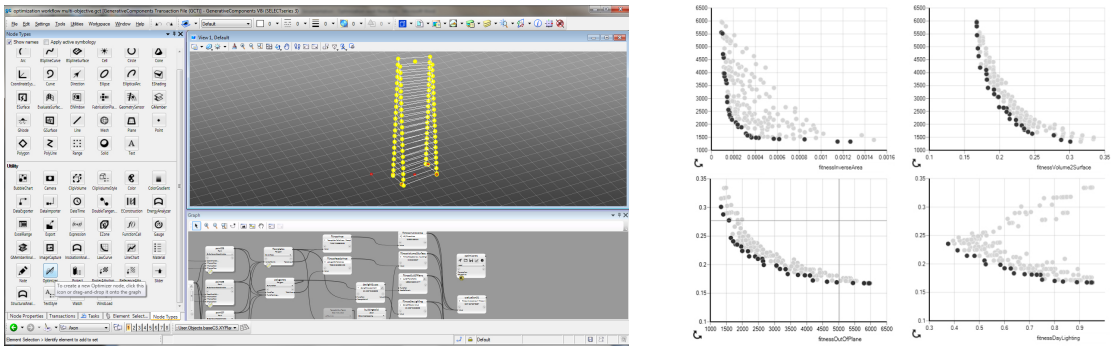
These components of *GC's* functionality come together to facilitate the current trend towards integration of analytical data and geometry. This process can essentially be reduced to the establishment of design parameters and an explicit model logic which is suitable for manual or automated configuration. Such configuration with regards to analytical performance objectives is described next.

2.3.2 Analysis, exploration, and optimisation

A positive and negative consequence of parametric models is the near-infinite number of alternative instances that can be easily generated. The issue is of selecting one, or a small set, of preferred solutions from the full potential set. This can be achieved through: i) human-selection (qualitative metric, experience, or aesthetics); ii) imposing constraints (removing invalid options); iii) or performance (quantitative analysis). In the third case, which is in focus here, analysing the performance is typically coupled with a goal-oriented search, or optimisation. The optimisation is fundamentally an automated selection process according to a specified criteria.

The elements of parametric CAD tools (as in *GC*, Figure 2.7a) mentioned previously make them conducive to such exploration and optimisation. Previously mentioned were two existing analysis tools that have been integrated with *GC*: *Bentley's STAAD.Pro* for structural performance; and the U.S. Department of Energy's *EnergyPlus* for dynamic thermal and energy simulation. These, or any other user-defined custom analysis, can be used for assessing the performance of a model.

An optimisation tool has also recently been integrated; the analytic framework is a single- or multi-objective genetic algorithm (GA). The GA is well-suited to complex design problems since it is stochastic (a random component to the search is better for complex state spaces) and population-based (the search results in a set of optimal individuals). For a single-objective problem, the search



(a) *GC* analysis framework screenshot (b) Typical Pareto fronts

Figure 2.7: Analysis framework setup and results in *GC* (Bentley Systems, 2013).

typically converges to a single solution; or a Pareto set for a multi-objective problem (Figure 2.7b). A core part of any optimisation procedure is the generation and evaluation of each individual's fitness, which is typically the most costly part of the whole process.

As well as the individual simulation time, which may range from seconds to hours, a large number of fitness calculations are generally required. As a solution to this, *STAAD.Pro*, *EnergyPlus*, and the analytic framework are currently running on the cloud alongside *GC* (Mueller et al., 2013; Mueller & Strobbe, 2013). The components of the entire analysis framework are shown in Figure 2.8. This setup gives a number of benefits but the primary one is process parallelisation, i.e. multiple individuals can be evaluated simultaneously. The cloud analysis tools can be used for single model simulations, rather than optimisation, although without any benefits of parallelisation. So whilst the framework is beneficial for optimisation cases by moving the computational cost to a parallel distributed system, it offers no benefits for single model evaluations.

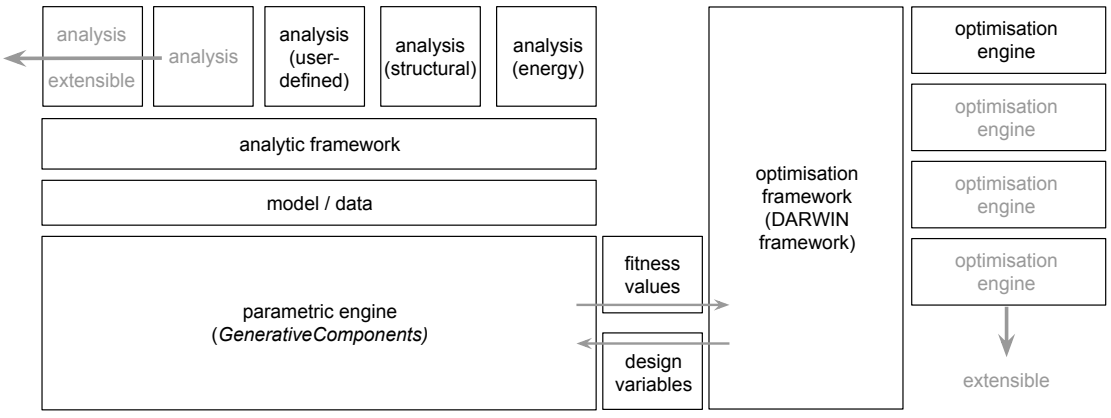


Figure 2.8: Cloud components of *Bentley's* analysis framework (Mueller & Strobbe, 2013).

Computational fluid dynamics (CFD) is currently not included as a component of the framework. This is due to two primary issues, both of which are systemic to CFD and will be discussed in later chapters: i) the cost of the simulation is generally greater than for structural or energy analysis, even with parallelisation; and ii) automation remains an issue, particular in mesh generation.

In summary, whilst the analytic framework does offer the ability for parallelisation required for optimisation, it does not solve the issue of individual costly simulations directly. This is most apparent with the current lack of a CFD solver, which represents perhaps one of the most complex analysis tools. The research presented in this thesis is formatted as a proposed development for integration with the analysis framework in §7.2.

2.4 Tool-User

PLP are an architecture practice currently working on a wide range of project scales and locations. The office is largely using *MicroStation* for 2D and 3D drafting, *Rhino* for 3D modelling, and *GC* or *Grasshopper* for parametric modelling. For early project stages such as competition and concept design, *Rhino* is used for initial sketch models whilst *GC* and *Grasshopper* tend to take their place after this. The following procession is typical: i) firstly, *Rhino* is very effective at directly translating ideas or paper sketches into 3D forms quickly and is usual for the first design draft; ii) secondly, a parametric model might next be constructed to either rationalise geometry, add complexity, tinker with parameters, or generate a family of models; and iii) after these, more detailed 2D and 3D representations are worked up in *MicroStation*. The second stage will be analysed in more detail here, especially with regards to tall building projects and analysis.

2.4.1 Parametric design

Building design is a complex (or complicated[3]) process which is often described as a wicked problem (originally proposed by Rittel & Webber (1973) for policy planning). They can be characterised by the following key points: i) there is no definitive formulation of a wicked problem; ii) there is no stopping rule; iii) solution success is both qualitative and quantitative; iv) there is no complete method to test a solution; v) they are always unique; and vi) there are an infinite number of possible solutions.

Each of these characteristics complicates the design process, however the creativity required to solve each problem often leads to unexpected solutions. When the problem is not wicked, for example with mass-manufactured warehouses, it is often not classified as architecture but rather engineering. Therefore the opportunities inherent in this type of problem can often be the source of its architectural value.

The most common approach to navigate these problems in practice is by exploring and assessing alternatives. It has been shown that experienced designers start by exploring a broad range of options initially and add depth, or detail, as time goes by. And that expert designers develop

[3]Complexity is an intrinsic property, whereas complication is extrinsic.

more alternatives than novices (Sheikholeslami, 2009), an exploration that is essentially a search through solutions (Simon, 1996). This kind of search is clearly strongly supported by computational methods and that by producing more alternatives, the end result can be of a better quality. Tools that enable this rapid generation of alternatives are called creativity support tools (Shneiderman, 2007).

An increase in the number of alternatives that are considered for a given project necessitates a selection process (§2.3.2). Design guidance offered by an automated selection or simply by analysis can filter the number of alternatives to a manageable size. However a simulation must be commensurate with the number of alternatives to be assessed and the risks associated with any subsequent decisions; leading to a tradeoff between speed and accuracy in simulation.

2.4.2 Speed versus accuracy

There is an inverse relationship between speed and accuracy in simulation, marked by the two extremes of slow/accurate and fast/inaccurate. The causes and implications of this are discussed in the literature review, but in this section the requirements of practice will be identified. However the basic concept is that for a given task or decision, higher accuracy requires more time and vice versa.

For the task of making a prediction about the aerodynamic behaviour of a given building, there are a variety of methods available. These range from experiential intuition as the fastest and least accurate, through to the slowest and most accurate computational simulations or experiments. The four existing identified categories are: i) knowledge-based; ii) numerical (visual basis); iii) numerical (physical basis); and iv) experimental.

It is logical to assign an appropriate tool for different design stages according to its allowances and requirements. At the start of a project speed is more important than accuracy, and vice versa towards the end. For generative design, the speed should be as near to instantaneous as possible, whilst the accuracy should be fitting with the risk associated with the consequences.

Quantifying an 'acceptable' level of accuracy is difficult for a number of reasons. One method is by comparison with other tools that are already accepted or being used in practice. A low-order CFD tool, the fast fluid dynamics (FFD) solver (Stam, 1999), has recently been used for a number of generative design problems (Chronis et al., 2012, 2011; Karagkouni et al., 2013; Wilkinson, 2011). It has also been suggested that it is suitable for early-stage design analysis (M. Jin et al., 2012, 2013; Zuo & Chen, 2009, 2010).

2.4.3 User survey

For each project stage, a survey of designers at *PLP* were asked to mark which level of accuracy was acceptable for simulation in general. Whilst the sample size was relatively small (15 people), the data given in Table 2.1 shows a clear increase of accuracy with project stage. For each accuracy band (e.g. 0-10%) the percentage of responses is given for each stage (e.g. 33.33% for Competition stage).

The last three project stages are roughly co-ordinated with the Royal Institute of British Architects' *Plan of Work* Stages 2: Concept Design, 3: Developed Design, and 4: Technical Design (RIBA, 2013). A 0% accuracy would indicate that the results are not important at all, but perhaps demonstrating technical capability. A 100% accuracy in this case would correspond to a maximum accuracy possible rather than in an absolute sense.

Table 2.1: Survey data on appropriate simulation accuracy versus project stage.

	Level of accuracy [%]									
Project stage	0-10	10-20	20-30	30-40	40-50	50-60	60-70	70-80	80-90	90-100
i) Competitions	**33.33**	13.33	6.67	20.00	0	6.67	13.33	6.67	0	0
ii) Concept design	0	6.67	20.00	13.33	**26.67**	6.67	6.67	13.33	6.67	0
iii) Development	0	0	0	0	0	13.33	**26.67**	20.00	13.33	26.67
iv) Detailed design	0	0	0	0	0	0	0	13.33	40.00	**46.67**

The mode responses are therefore: Competition 0-10% (5/15 votes); Concept design 40-50% (4/15 votes); Development 60-70% (4/15 votes); and Detailed design 90-100% (7/15 votes).

Since the target application for this research are the first two stages, Competition and Concept Design, the survey data is roughly suggesting an acceptable accuracy of between 0 and 50%. The first observation is that the responses for accuracy have a linear relation to project stage, from 0% at the start to 100% at the end. This highlights a complication in the survey question: beyond general terms, what does it exactly mean to have 0 or 100% accuracy. The answer must depend on against what the accuracy is measured since even a random guess would likely exhibit some 'accuracy' simply by chance. Whilst the conclusions that can be drawn from the survey are limited, it helps to frame the validation from Chapter 4 onwards, and to understand the expectations of designers.

2.5 Summary

The purpose of this chapter was to describe the motivation for this work from the perspective of a typological design problem, software development, and requirements of the software user. Firstly, the challenges of generative tall building design when paying consideration to aerodynamic wind effects is focused on the availability of analysis tools with suitable speed and accuracy characteristics. This secondly leads to the ongoing work of the tool developer, *Bentley*, and their integration

of *GC* with an online analysis and optimisation framework. A response to the further development of the thesis for this purpose is detailed in Chapter 7. And lastly, the early project stage requirements of the tool-user, *PLP*, specific to the speed-accuracy tradeoff in simulation are described, as well as their response to a short survey into determining an appropriate degree of accuracy. It is established that the objectives of both *PLP* and *Bentley* coincide towards the development of fast-yet-accurate wind analysis tools for generative tall building design; and that together, these three components essentially form the motivation and brief for the subsequent chapters.

Chapter 3

Literature Review

The review is structured in three parts that progress from a practical design problem through to methodological solution. The three parts respectively relate to design context, simulation, and solution:

§3.1 Generative Design: on trends towards parametric computational analysis; §3.2 Fluid Simulation: on the range of approaches to computational fluid dynamics, their speed and accuracy, and assessment through validation; §3.3 Machine Learning: on approximation or reduction of performance models for design, namely on computational fluid dynamics.

As is common with interdisciplinary research (here between architecture, wind engineering, fluid simulation, and computer science), each of these parts is essentially a large research field in its own right. This necessitates a relatively broad review focusing on establishing the key trends, limitations, and approaches in relevant research.

3.1 Generative Design

Early project stages (concept phase, RIBA (2013) stage 2) commonly involve sizing, topology, positioning, orientation, and large-scale form or massing decisions (Samareh, 2001). Decisions made at this time are used as the basis for subsequent stages (development and technical phases, RIBA (2013) stages 3 and 4). Although the concept design stage only accounts for 15% of the total design fee, about 80% of the resources required in construction are committed by decisions made at this point (Park et al., 2004). It is therefore the most critical project stage where a broad range of potential design alternatives are created and evaluated before moving forwards.

Early design assessments are mostly based on consultant's expertise and experiential knowledge, with a lack of simulation tools tailored to facilitate quick and accurate analysis. de Wilde et al. (2001) analyse 70 energy-efficient building projects, finding that performance simulation at early stages was largely absent in most cases. Most often, consultants become involved at later stages by which time they are restricted to a narrower 'design option space', in turn limiting the impact of

the analysis and subsequent recommendations. This problem is even greater, and advanced tools and expertise more in need, for experimental projects that are towards the frontier of design and technology, such as tall buildings (Augenbroe, 2003).

Parametric CAD tools, such as *GenerativeComponents* (Bentley Systems, 2013) and *Grasshopper* (Davidson, 2013) plug-in for *Rhino* (McNeel, 2013), can facilitate the creation of more alternatives relatively quickly. Both of these tools are targeted towards early-stage architectural design and have generally similar characteristics (§2.3.1). Interest in parametric work practice has been renewed in the last decade with the introduction of these two tools (in 2003 and 2007 respectively), along with an interest in coding and mathematical rationalisation (Peters & Peters, 2013). Whilst the creation of alternatives with parametric, or generative rule-based, models has become relatively mainstream now, the incorporation of analysis data is still ongoing and presents an interesting set of challenges.

3.1.1 Integrating analysis

The shift of simulation tools from pure analysis to design aid is being driven by advances in computational resources (hardware) and from developers (software). These trends have been noted or pursued by many, exemplified in a collection of articles by practitioners and academics edited by Kolarevic & Malkawi (2005), and in a seminal review article by Malkawi (2004). Malkawi suggests that digital simulation tools can be used to support performance-driven design through optimisation, either manually (exploration) or automatically (search algorithms).

Whilst this is now basically feasible with energy/thermal, structural, solar simulations etc, it is still largely beyond the possibilities of more intensive simulations such as computational fluid dynamics (CFD). This is due to the simulation cost (the time to setup, run, and extract data) and that CFD is still an expert tool, requiring knowledge to configure models and interpret results. Both of these issues with integrating CFD will be examined in more depth:

- *Expertise:* With increased availability, attractiveness, and awareness of simulation tools, architects have begun to use them to some extent in practice. It marks a change from their sole use by academics or engineering consultancies, i.e. self-proclaimed 'experts', and has been in response to broader trends in architecture towards sustainability, performative, and computational design. There is concern however that the tools should only be used by those with the correct expertise and specialist knowledge. The reasons for this concern are not entirely unjustified, but they largely stem from a defensive position where tool requirements and application are quite different from architectural practice. One justification is that simulation results are dependent on the many input configuration parameters (such as boundary conditions, turbulence models, convergence criteria, mesh quality, geometric representation,

domain size, etc.) (Tominaga et al., 2008).

From this perspective, where possible tools should be developed to be simpler to use, more intuitive, and more robust to input parameters. For example, meshing is one of the most important steps in preparing a simulation and involves a relatively complex set of compromises or tradeoffs. Automatic domain mesh generation has been a major development in CFD and should still require sensitivity studies to determine result independence. A solution for 'non-expert' users could be a simpler interface that requires the user to only specify the desired simulation time, resolution (ability based on computational resources), or accuracy (automatic problem-type identification and comparison with validated models).

On the contrary though, designers are now engaging with these existing tools and will continue to do so, regardless of whether they consider themselves experts or not. The concerns may also not take into consideration the applications that architects are involved with and the fundamentally different requirements. At early stages and for generative design, comparative results for gauging the relative difference between options allows for strategic decisions and improvements. It is only at later stages that quantitatively definitive data are necessary (Lawson, 2006; Lu et al., 1991).

- *Speed:* In cases such as CFD, the speed (or inversely, the response/evaluation/run time) can become prohibitive, obstructive, or too costly for inherently iterative generative design. Therefore, previous attempts to fit CFD to early stages by increasing speed, tend to compromise by simplifying either: i) geometry; ii) meshing; or iii) solver. Such simplifications, or solution approximations, all succeed in increasing speed but at the cost of a substantial loss of accuracy or realism:

  Geometry: Model geometry may be reduced to simple orthogonal shapes and complex boundary conditions avoided. CFD has been integrated with a genetic algorithm (GA) to optimise indoor environmental conditions, (Malkawi et al., 2005, 2003). In these models, simple orthogonal structured meshes can be used which are both simple to recreate with changing configurations and fast to run. For certain classes of problems it can be appropriate to use 2-D simulations, namely when it is clear that the flow perpendicular to the analysis plane is minimal or of no interest. The primary example of this is airfoil analysis and optimisation with either GAs (Duvigneau & Visonneau, 2003; Giannakoglou, 2002; Marco et al., 1999; Shahrokhi & Jahangirian, 2007) or gradient descent (Elliott & Peraire, 1996; Huyse, 2001; Huyse & Lewis, 2001; Zingg & Elias, 2006). However in external turbulent flows, such as around buildings, it is typically only appropriate to use 3-D simulations.

  2-D CFD simulations can also be used for optimisation of tall building topology for aerodynamic performance, by taking a small number of horizontal section profiles at various heights

(Kareem et al., 2013). The proposed methodology had not been fully tested so it is unclear whether it will be successful or not, however the simulation of the flow in 2-D sections is a relatively large simplification, especially for tall buildings where down- and up-drafts can occur.

Mesh: Creation of surface and volume meshes (spatial discretisation) typically requires a degree of user input, which restricts the degree of automation possible. The speed and accuracy of a simulation are strongly dependent on the extent, type, and resolution of the mesh. Meshes can be generally classified as either structured or unstructured: structured meshes are basically orthogonal and hexahedral (cuboids or voxels) where the physical location of the nodes are related to the mesh, i.e. $(x, y, z) = f(i, j, k)$. This implies that a node's neighbours are easy to find, i.e i's neighbours are $i - 1, i + 1$ etc.; or unstructured such as tetrahedral, where each node's physical location and neighbourhood connectivity needs to be stored. Structured meshes are faster and simpler to implement, but are not suitable for non-orthogonal complex geometry. If structured meshes are used on non-orthogonal geometry, the result is often a severe approximation of the original form (Figure 3.1).

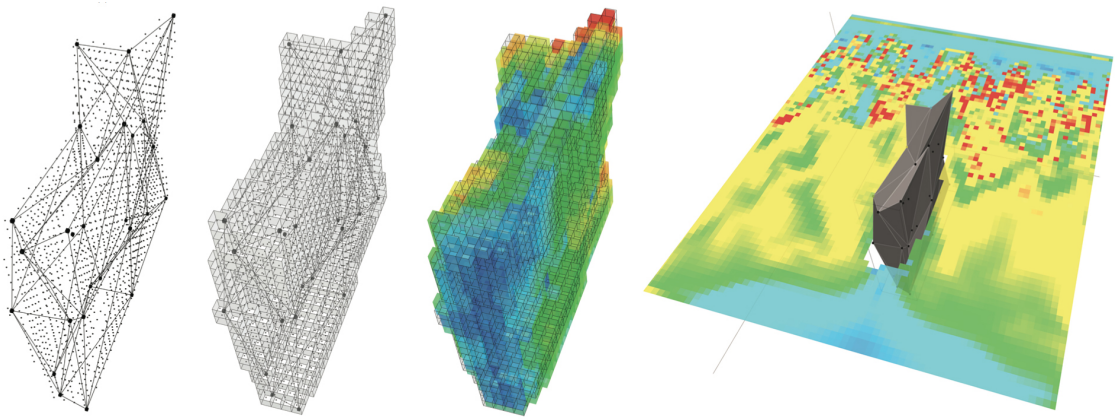


Figure 3.1: Example of a structured mesh with a low-order CFD solver (Wilkinson, 2011).

In this example, the figure shows (from left to right): barycentric model sampling; structured mesh; with surface pressure; and field velocity at a horizontal section plane and the original surface mesh geometry. The combination of the low-resolution structured mesh and the low-order CFD solver (§3.2.3) meant it was possible to run extensive single- and probabilistic multi-conditional shape optimisation with a GA (Wilkinson, 2011). A similar study preceded this where the shape of a 2-D canopy surface was optimised to reduce wind loads with a single-objective GA (Chronis et al., 2011).

Solver: The vast variety of CFD solver approaches suit a range of speeds, accuracies, and applications (§3.2). They can generally be classified by their spatial or temporal treatment of turbulence.

The conventional compromise or tradeoff between speed and accuracy occurs in most simulation tools and will be discussed next.

3.1.2 Speed-accuracy tradeoffs

By considering the accuracy and speed of various CFD simulation tools, a spectrum of approaches can be observed. A neurological model of decision-making in animals can be used as an analogy to this tradeoff. Chittka et al. (2003, 2009) investigated the behaviour of bees when confronted with the task of collecting nectar from flowers. They found that for low-risk tasks (e.g. collecting nectar) the bees opted to make faster decisions with a lower accuracy, i.e. choosing to 'guess' a solution quickly; and conversely for high-risk tasks (e.g. identifying predators), slower decisions resulted in higher accuracy. These two behavioural extremes are either 'impulsive' or 'reflective'.

The range of wind analysis methods can be notionally positioned on a speed-accuracy tradeoff graph (Figure 3.2b). These are grouped as: knowledge-based; low-order simulation; high-order simulation; and experimental. It becomes clear in this context that by investing more time in the decision/simulation then accuracy can be improved, and vice versa.

If the task changes in complexity the tradeoff changes too (Figure 3.2a). For a simple task (black curve) the tradeoff will not be apparent since high accuracy can be achieved for any decision time. The level of experience or skill with which the task is undertaken also affects the tradeoff, decreasing time as skill increases.

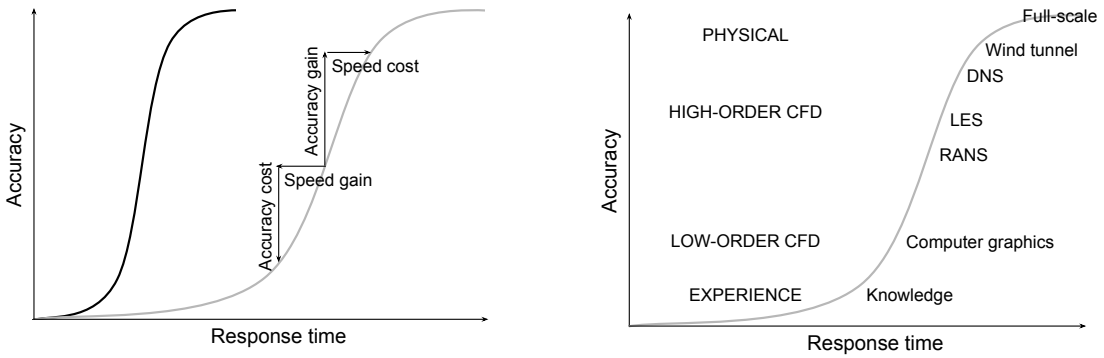


Figure 3.2: Speed-accuracy tradeoffs: (a) notional speed-accuracy tradeoff for simple (black) and complex (grey) tasks (Chittka et al., 2009); (b) range of fluid analysis methods.

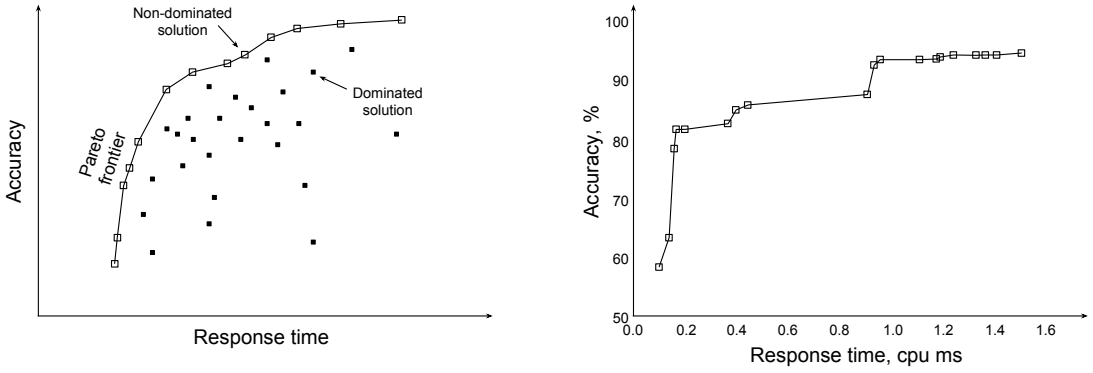
Burns (2005) extends the findings of Chittka et al. (2003) by demonstrating that the fast-inaccurate bees collect nectar at a higher rate than the slow-accurate ones. It is also argued that the focus on accuracy alone can lead to sub-optimal behaviour overall, i.e. starvation. This highlights that success should not necessarily be measured by the immediate task (identifying the correct flower, or simulation accuracy), but to the broader one (of survival, or designing a well-performing building).

The analogy of natural speed-accuracy tradeoffs has been used as a parallel to the accuracy and response time of design analysis tools (Chronis et al., 2012). However, the analogy is only useful as a concept to understand the spectrum of simulation approaches and does not explicitly take into account suboptimal approaches.

3.1.3 Pareto frontiers

Tradeoffs can also be presented as a Pareto frontier, as in multi-objective optimisation, where the method is represented in the objective space (here a 2-dimensional space of time and accuracy). In this way, the approaches are classed as either a dominated or non-dominated solution, where the non-dominated set forms the Pareto frontier (Figure 3.3a). This allows for the introduction of suboptimal solutions, or ones that are not Pareto efficient in terms of speed or accuracy.

This has been used by Lu et al. (1991); Tcheng et al. (1989); Yerramareddy et al. (1992) to show a range of performance models with varying speeds and accuracies. By using machine learning, they were able to create performance models that approximated a higher-order model of a combustion engine. This gives the designer or engineer the freedom to select an appropriate performance model at different project stages.



(a) (b)

Figure 3.3: Speed-accuracy objective space: (a) notional Pareto frontiers; (b) non-dominated generated performance models (Lu et al., 1991).

Traditionally, CFD originated in the aerospace engineering field, and has since the 1980s (C. J. Baker, 2007; Murakami, 1997; Selvam, 2008) been increasingly of interest in the built environment. In this time it has increased in use, accuracy, and application; migrating from analysis to design guidance and infiltration of architectural practice. However, the origins of the tools have different speed-accuracy requirements meaning the migration is partially flawed.

For the original aerospace applications (e.g. spacecraft re-entry at 17,500mph) where there is the potential for serious failure and loss of life, accuracy is of paramount importance and the margin of error low. Whilst these dangers are still possible for buildings, the margin of error is greater due

to the redundancy inherent within such heavy structures as compared to smaller, lighter aircraft. Subsequently the tools necessarily took on a fundamentally analytical role to achieve the required accuracy at the cost of speed (i.e. slow-accurate); this is in contrast to the design guidance role here, where in reality fast-yet-accurate tools are required.

3.2 Fluid Simulation

Applications of computational fluid dynamics (CFD) in the built environment can be arranged roughly by scale of analysis: i) around a human body ($\sim 1m$); ii) internal building flow ($\sim$1-10m); iii) external flow around a building ($\sim$10-100m); iv) pollution dispersal / pedestrian wind comfort ($\sim$100-1000m); v) city or regional climates ($\sim$>1000m). The application and scale of concern here is the external wind flow around tall buildings. In this section, characteristics of the flow type are introduced, followed by an introduction to available CFD approaches, their applicability and validation.

3.2.1 Bluff body flow

The boundary layer near to the earth's surface can extend in depth up to between 300 and 600m, therefore building aerodynamics are intrinsically related to the movement of air in this boundary layer. Its structure or vertical distribution, i.e. velocity, turbulence intensity and length scale, is determined strongly by the surface roughness (Cermak, 1976). The vertical distribution of wind speed in its most basic form can be modelled as a power law:

$$U_x = U_r \cdot \left(\frac{Z_x}{Z_r}\right)^{\alpha} \tag{3.1}$$

where U_r is a reference wind speed at a reference height Z_r, and the exponent α is a function of both the atmospheric stability and surface characteristics. For smooth terrain on land the following equation by Panofsky & Dutton (1984) can be used to calculate α at a height z=10m and a roughness length z_0=1cm:

$$\alpha = \left[ln \cdot \left(\frac{z}{z_0}\right)\right]^{-1} \tag{3.2}$$

giving a value of 0.14. The reasoning behind the use of a neutral wind profile in the simulations is given in §4.2.2. The equation is validated by Hsu et al. (1994) who confirmed experimentally that under near-neutral atmospheric stability conditions at open sea the mean and standard deviation for α was 0.11 $\pm$ 0.03, compared with a calculated 0.10.

A bluff body is a term used to describe the shape of an object in relation to the fluid behaviour around it, characterised by a dominance of pressure drag rather than frictional (viscous) drag. For

streamlined objects the flow is attached to the surface (i.e, there is no separation such as for an airfoil with a small angle of attack) and the frictional drag is related to the surface area. However for bluff bodies with separated flows, the pressure drag is related to the shape and cross-sectional area. The majority of buildings and structures therefore fall into this category, causing some of the challenges in simulating wind flow around buildings. The presence of sharp edges common to buildings cause separation, reattachment, and vortices, i.e. spatially and temporally unsteady turbulence, making it difficult to resolve numerically (Murakami, 1997).

The separation caused by shape, high Reynolds number, or sharp edges leads to the formation of wake vortices. Figure 3.4 shows vortex development behind a cylinder for increasing Reynold's number (air speed). The oscillation of these vortices, and the behaviour known as vortex shedding or excitation, can create large cross-wind loads, or dynamic response, horizontally perpendicular to the flow direction. In a study exploring the effect of building shape on wind-induced structural response through the wind-tunnel testing of a small set of basic building shapes (square, rectangular, elliptic, circular, and triangular), it was concluded that by considering building shape, wind performance can be fundamentally improved (Gu, 2009; Merrick & Bitsuamlak, 2009). Both seek to encourage designers to explore shape effects with regards to bluff body aerodynamics at early stage design.

The Reynold's number (Re) is a dimensionless ratio of inertial to viscous forces, therefore it can be used to characterise laminar or turbulent flows:

$$Re = \frac{Inertia force}{Viscous force} = \frac{\rho \cdot V \cdot L}{\mu} \tag{3.3}$$

where: ρ is the fluid density $[kg \cdot m^{-3}]$; V is the velocity $[m \cdot s^{-1}]$; L the length scale $[m]$; and μ the dynamic viscosity $[Pa \cdot s]$. Since the fluid density (incompressible), length scale, and dynamic viscosity remain constant, Re is largely dependent on the velocity V.

In our case here, for a general idea of the Reynold's number: if $\rho = 1.2kg/m^3$, $V = 10m/s$, $L = 20m$, and $\mu = 1.8e^{-5}Pa.s$; then Re is of the order of magnitude of $13e^6$, and within the turbulent regime.

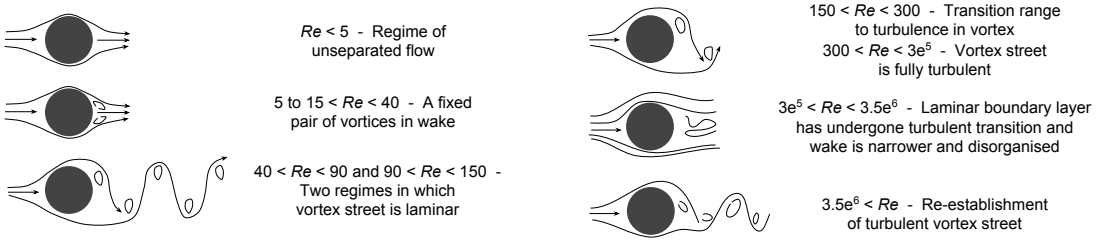


Figure 3.4: Vortex development around a cylinder cross-section at various Reynolds numbers.

Even under steady boundary conditions, flow separation and turbulence can cause unsteady fluid

behaviour (the terms 'temporal', 'transient', or 'time-dependent' are used interchangeably); a fundamental characteristic of bluff bodies such as buildings. In the following sections, the advantages and disadvantages of steady-state (RANS) and transient simulations (LES) are discussed.

3.2.2 Urban interference

Interference refers to the increased or decreased effect that adjacent buildings may have upon the wind behaviour of one another. Within an urban scenario this is very common, and since the effects can be significant it is usually necessary to consider the context within the simulation. That is, independently designed buildings can not be treated in isolation but should be seen within their context. There is a common misconception that interference always reduces wind loads from the isolated case, which may be true for a uniformed array of similar buildings, however wind loads can be increased in more complex, realistic case.

Along with the large fields of bluff bodies and computational wind engineering, interference is almost as broad in itself. Since the collapse of three out of eight cooling towers at Ferrybridge power station during a storm in 1965, significant experimental research led to advances in the theoretical understanding and design guidance concerning wind interference (Ford, 1994). Today, the ongoing body of work is especially concerned with creating generalised recommendations, a difficult issue due to the huge variation in potential scenarios. The key factors in determining the effects of interference are the size, shape, and configuration of the buildings with respect to the direction of flow. The effects have been shown to be as great as up to 46% under-prediction and 525% over-prediction from code specified loads on simple prismatic buildings. An over-prediction is clearly far less dangerous than an under-prediction, the difference being between structural safety or comfort, and over-engineered, inefficient structures. A thorough review of the full history and state of interference can be found by Khanduri et al. (1998), and a summary of more recent typical studies can be found in Table 3.1.

Table 3.1: Summary of existing interference global parameter sensitivity studies.

No.	Evaluation method	O.	SD.	AR.	C.	α	Source
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Lee & Fowler (1975)
2	WT		$\bullet_{X,Y}$	$\bullet$		-	Sakamoto & Haniu (1988)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Taniike (1992)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Saunders & Melbourne (1979)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Bailey & Kwok (1985)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Taniike & Inaoka (1988)
2	WT	$\bullet$	$\bullet_{X,Y}$	$\bullet_{Z}$		0.19	Agrawal et al. (2012)
2	WT + CFD (RNG k-ϵ)	$\bullet$				0.16	A. Zhang & Gu (2008)
2	WT		$\bullet_{X,Y}$	$\bullet_{X,Y}$		0.14	Taniike & Inaoka (1988)
2 & 3	WT		$\bullet_{X,Y}$	$\bullet_{X,Z}$		0.16	Z.-N. Xie & Gu (2004), Gu & Xie (2011)
5	WT	$\bullet$	$\bullet$		$\bullet$	$\bullet$	Lam et al. (2011)
5	WT	$\bullet$		$\bullet_{X,Y}$		0.15	Jianguang (2008)
Multi.	CFD (RNG k-ϵ)	$\bullet$			$\bullet$	0.22	A. Zhang et al. (2005)

$\bullet$ varied in study; - no data; WT wind-tunnel; **No.** Number of Study Buildings; **O.** Orientation; **SD.** Separation Distance; **AR.** Aspect Ratio; **C.** Configuration; α Wind profile exponent; X is direction perpendicular to flow; Y stream-wise; and Z vertical.

In all the cases shown in Table 3.1, simple cuboids were used and variable parameters were height, separation distance, and aspect ratio; in other words, a sensitivity analysis around the 2-D horizontal plane. There have been no studies considering realistically complex shapes or contexts, probably because the knowledge attained in evaluating them is typically esoteric and difficult to generalise. The selection shown in Table 3.1 is representative, rather than exhaustive, of existing studies. Further examples are given in the following section where machine learning is applied to attempt generalisation along the same lines as these sensitivity analyses.

3.2.3 Computational fluid dynamics (CFD)

Bluff body flows and interference are both fluid problems that are addressed through computational analysis. As an alternative to wind-tunnel or even full-scale testing, the use of simulation has risen dramatically with the increase in computing power in recent decades. The dominant view is that at its current state of development, CFD should be used in conjunction with wind-tunnel experiments for validation, but that use for preliminary assessments is acceptable (Bitsuamlak, 2006; Dagnew et al., 2009; Selvam, 2008; Stathopoulos, 1997). Traditionally, the almost singular focus on accuracy with secondary consideration of simulation time in the research community is at odds with practitioners. For instance, it is generally suggested that CFD can be used for approximate assessments to provide insight into preliminary design scenarios and that whilst CFD is *'definitely a good friend of wind engineering, it has not yet become a true ally'* (Stathopoulos, 1997). Conservative views, such as Cochran & Derickson (2011), place particular concern on impressive graphics creating a misleading idea of accuracy and concerns of the complexities of flow separation and reattachment around bluff bodies being beyond numerical capabilities.

Nonetheless, to solve these fluid problems a range of simulation approaches are available, fitting into one of two classes: Eulerian (mesh-based) and Lagrangian (particle-based). Eulerian includes the Fast Fluid Dynamics (FFD), Reynolds-Averaged Navier Stokes (RANS), Large Eddy Simulation (LES), and Direct Numerical Simulation (DNS); the Lagrangian includes Smoothed Particle Hydrodynamics (SPH), Discrete Vortex Method (DVM), and Lattice Boltzmann Method (LBM). The focus in this section is on the Eulerian set, however Lagrangian methods will also be briefly introduced.

Notionally in descending order of accuracy and increasing speed, the DNS, LES, RANS, and FFD approaches will be briefly introduced. Whilst RANS, LES, and DNS (Figure 3.5) are broadly the conventional approaches to CFD (although each with different characteristics), the FFD is non-conventional in the sense that it is a relatively recent development. The three conventional methods can be generally classified by the turbulence length scales that are either solved directly or modelled. For instance, DNS directly solves all scales down to the smallest Kolmogorov dissipation

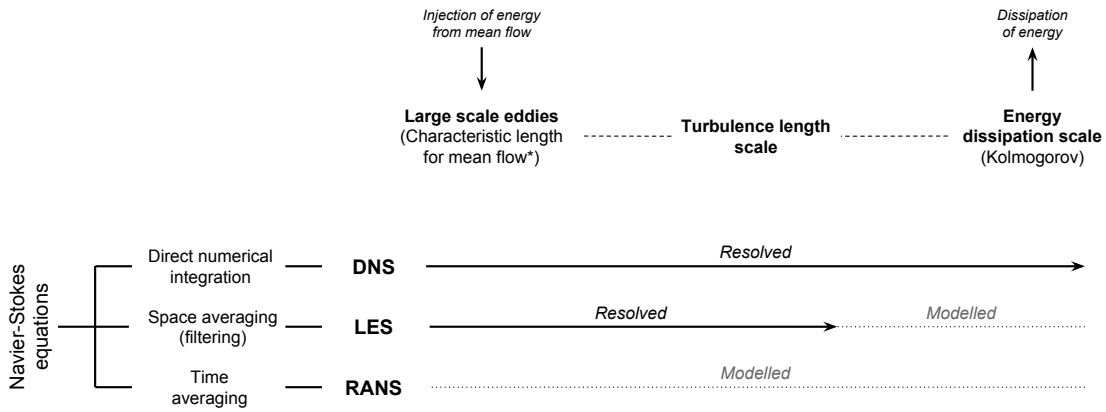


Figure 3.5: Simulation approaches and turbulence models (Davoudabadi, 2012; Murakami, 1997).

scale ($\eta = [v^3/\epsilon]^{\frac{1}{4}}$, where ϵ is the dissipation of turbulent kinetic energy per unit mass), whilst RANS applies closure models to all turbulence lengths below the characteristic length.

The temporal behaviour of a simple fluid simulation is shown in Figure 3.6 comparing DNS, LES, and RANS approaches. Time-averaging, or steady-state, RANS is invariant to both fluctuations and periodic flow behaviour; LES gives the periodic behaviour but not fluctuations; and DNS is similar to experimental data in its modelling of both periodic and fluctuating flow components.

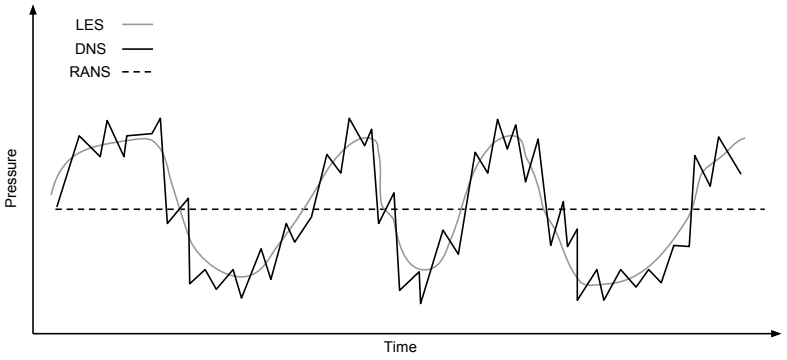
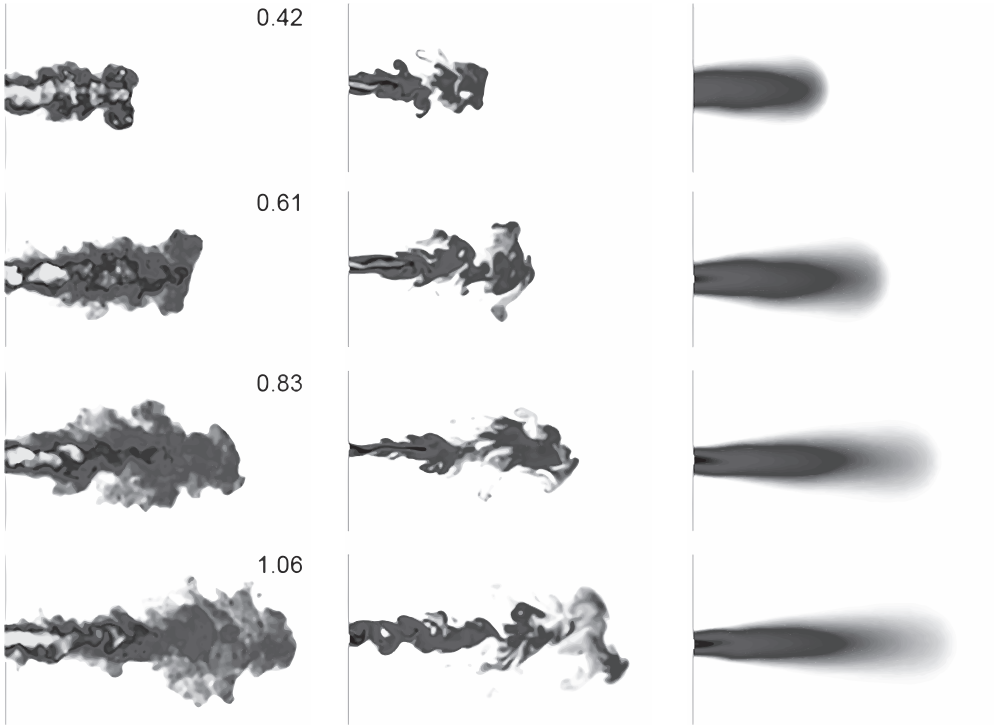


Figure 3.6: Transient turbulent flow comparison.

The spatial behaviour of a turbulent jet mixing flow is shown in Figure 3.7 qualitatively comparing experimental, LES, and RANS results at four time-steps. Som et al. (2012) used a mesh resolution of $0.5mm$ ($\sim$0.3million cells) for the RANS and $0.125mm$ ($\sim$2.0million cells) for the LES. The experimental data was obtained by Rayleigh scattering imaging, an approach to gain 2-D and 3-D plots of molecular density through measurement of laser scattering. The RANS and LES both predict the dispersion fairly well however there are *'marked differences'* in the spray structure between the two simulations; the RANS predicting smooth, averaged distributions; and the LES capturing the instantaneous structure well. Computation time for the high resolution LES was 8-10 days (on a 20-core machine) compared with only 80 hours (on an 8-core machine); a factor of

between 2.4 and 3.0 times faster.

The basis of most CFD solvers, and other finite element methods (FEM) for physical simulations using partial differential equations (PDEs), is spatial and temporal discretisation via meshing and solver time-steps. The solution approximation accuracy is therefore dependent on the discretisation resolution, itself dependent on user choice and available computational resources. Meshing involves reducing the computational domain from a continuous region to a finite set of geometrically simple and bounded elements (typically triangles, quadrilaterals, tetrahedra etc). Selection of an appropriate mesh is of importance and should consider the size of the domain, complexity of the geometry, the focus of the study, and what is deemed a suitable resolution for the study. There is therefore a relatively high degree of experience required to generate the best mesh for the particular fluid flow problem.



(a) Experiment (b) LES (c) RANS

Figure 3.7: Spatial turbulent flow mixing comparison (Som et al., 2012).

This discretisation process turns the PDEs into difference equations, or from continuous rates of change into discrete finite ones that can be solved computationally (Winsberg, 2008). The main pre-requisite for this in CFD is mesh generation and specifying the simulation time-step interval. Inherent in any discretisation is the approximation of the original continuous geometry, meaning that by controlling the discretisation resolution it is possible to minimise the inaccuracy in the approximation to a certain extent. Using a fine mesh, however, has limitations in terms of computational resources and time, where geometric complexity and simulation at a desired resolution requires meshes that are impractically fine (Frey & George, 2010).

For incompressible Newtonian flow, the Navier-Stokes equations can be written as:

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v}\cdot\bigtriangledown\mathbf{v}\right) = -\bigtriangledown p + \mu\bigtriangledown^2\mathbf{v} + \mathbf{f} \tag{3.4}$$

Where $\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v}\cdot\bigtriangledown\mathbf{v}\right)$ are the inertial forces, and $-\bigtriangledown p + \mu\bigtriangledown^2\mathbf{v}$ is the stress divergence. The separate components are: the unsteady acceleration $\frac{\partial \mathbf{v}}{\partial t}$, the convective acceleration $\mathbf{v}\cdot\bigtriangledown\mathbf{v}$, the pressure gradient $-\bigtriangledown p$, the viscosity $\mu\bigtriangledown^2\mathbf{v}$, and other body forces $\mathbf{f}$ (such as gravity). It is the convective acceleration component that is the source of nonlinearity in all fluid flows. This nonlinearity leads to turbulence, specifically when the Reynolds number (the ratio of inertial forces to viscous forces) reaches a threshold transitional region. As mentioned previously, CFD approaches can be broadly classified by their treatment of turbulence and will be discussed below.

Direct Numerical Simulation (DNS)

DNS computes turbulent flow by directly solving the Navier-Stokes equations without any approximative sub-grid models, requiring a very fine mesh resolution to capture the smallest eddies in the turbulent flow at very small time steps, even for a steady-state flow (Chen & Zhai, 2003). Since turbulence exhibits a wide range of spatial and temporal length scales, bluff body flows are currently not feasible to compute with DNS. For this reason, complex geometry and high Reynolds number (turbulent) flows are beyond current capabilities, with even simple orthogonal 3-D shapes necessitating substantial computational resources and time (Moin & Mahesh, 1998). DNS can be used, however, for experimental research alongside physical modelling as a tool for developing better turbulence models for use with RANS and LES.

Large Eddy Simulation (LES)

LES (Deardorff, 1970) separates turbulent motion into large and small eddies, computing the large eddies in a three-dimensional and transient way while estimating or filtering the small eddies with a subgrid-scale model. When the grid size is sufficiently small, the impact of the subgrid-scale models on the flow motion is negligible; that is, the subgrid-scale models tends to be universal because turbulent flow at a very small scale seems to be isotropic (Chen & Zhai, 2003). LES is therefore more suitable than RANS for modelling wind flow around buildings as it can capture turbulent flow more accurately (Eggenspieler & Menter, 2012). It has been shown to cope well with complex, large-scale vortex-shedding as is common with tall buildings, and to generally be a more faithful approximation of turbulent fluid motion (Ochoa & Fueyo, 2004; Rodi, 1993, 1997; T. Tamura et al., 2008; Y. Tamura, 2009). However its longer simulation time makes it prohibitive for exploring multiple designs.

Reynolds-averaged Navier-Stokes (RANS)

Reynolds-averaged Navier-Stokes (RANS) is by far the most commonly used approach in practice. It solves the statistically averaged Navier-Stokes equations by using turbulence transport models to simplify the calculation of the turbulence effect. The approximation effectively smooths out the transient and spatial aspects of turbulence, but returns significant improvements in speed (Chen & Zhai, 2003). This compromise of lower accuracy and greater speed means that more of the system can be modelled rather than smaller components, as well as multiple designs.

The use of Reynolds-averaged Navier-Stokes (RANS) with a k-ϵ turbulence model is deemed the most robust method for commercial software given the broad range of scenarios being simulated in practice (Laurence & Mattei, 1993). RANS is time-averaging, whereby an instantaneous quantity is processed through Reynolds decomposition to separate the steady (time-independent) and fluctuating (time-dependent) components of the flow (Figure 3.6). This decomposition separates out a nonlinear part, the Reynolds stress, which is the cause of turbulence, at which point a separate turbulence model (such as k-ϵ) can be applied. This means that turbulence occurring at smaller sub-grid scales can be dealt with by the solver separately, avoiding the need to generate an overly fine mesh and requiring less computation time than LES or DNS (A. Zhang & Gu, 2008). The subsequent limitation is that it fails to accurately capture flow changes with a time component, like oscillating vortex shedding.

Fast Fluid Dynamics (FFD)

Existing research into fast computational fluid dynamics stems from the computer graphics industry, primarily for video games and movie special effects. In both areas there is a demand for fast algorithms to approximate fluid behaviour such as smoke, water or any particulates, with a visual- rather than physical-based accuracy (Wicke et al., 2009). The distinction between FFD and CFD is drawn from the original intended applications. Efforts have been focused on three fluid approximation approaches: mesh-based Eulerian; mesh-less Lagrangian; and model reduction. Within Eulerian approaches, a significant advancement was made by Stam (1999) who developed an unconditionally stable advection step, culminating in the *Stable Fluids* FFD solver that allows for real-time fluid simulations. Further development were proposed by Feldman et al. (2005) and Elcott et al. (2007), however with a focus on accuracy the speed reduced considerably to below real-time.

With speed as priority with the FFD, simple and low order schemes are used: such as linear interpolation in a semi-Lagrangian approach rather than higher order non-linear interpolation, and the simplest pressure-correction projection method available (Zuo & Chen, 2009). The low order

discretisation (first-order for time and second-order for space) have been seen to generate too much numerical dissipation (Fedkiw et al., 2001); this is less of a problem for the intended application of video games, rather than wind engineering, as action movement can add energy to the flow and keep it moving in a believable way. However, despite these substantial solver approximations, there has been interest lately on whether these methods can be used for more practical applications, such as real-time smoke dispersal for evacuation planning and wind engineering.

Lagrangian simulation

As mentioned at the start of this section, Lagrangian or particle-based simulations are fundamentally in contrast to the Eulerian approaches. Whilst the Eulerian system uses a fixed finite element or volume method and is conceptually field-based, Lagrangian systems use collections of moving points and is mesh-free. This includes Smoothed particle hydrodynamics (SPH), Discrete vortex method (DVM), and the Lattice-Boltzmann method (LBM). Instead of mesh resolution, higher accuracy is achieved through large numbers of particles which necessitates an additional calculation for surface interactions.

Although these particle-based tools are reasonably established and show potential, typical applications remain within computer graphics and rigorous validation studies are scarce. They have therefore not been accepted by the traditional computational wind engineering community.

3.2.4 Geometric complexity

The level of detail, or resolution, of a model can be measured by the scale of the smallest element. As a project progresses from early massing and form decisions to detailed design the level of detail typically increases. This progression is associated with the risk involved in the consequence of decisions, as well as traditional work practices.

Level of detail can be simply seen as surface roughness, such as the addition of balconies, louvres, glazing frames / mullions, structural components etc. Whilst it is typical for early stage models to be absent of this level of detail, it is significant to understand the effects that adding detail has on aerodynamic behaviour. There is agreement in the literature that surface roughness affects the wind-induced pressure on buildings, either positively or negatively.

In a study comparing experimental and numerical results from flow over shark skin (D.-y. Zhang et al., 2011), it is shown that the micro-structure (saw-teeth with width=$150\mu m$ and length=$250\mu m$) present on the skin control the occurrence of turbulence and effectively reduce surface friction, in a similar way to golf-ball dimples. The structure of the skin is shown in Figure 3.8; from left to right, the 3-D model of a single scale derived from a scanning electron microscope (SEM), a rationalised

single scale laid out in the macro-structure pattern, and the DNS flow field focused around a single scale. As well as a DNS study primarily for flow visualisation, experimental tests also found that reductions of between 8.72% (at 5.5 $m \cdot s^{-1}$) and 12.81% (at 3.3 $m \cdot s^{-1}$) can be made on a similar smooth surface. Further improvements in friction reduction is noted when sharks travel very fast, brought on by the release of mucus (a type of polymer) acting as a lubricant.

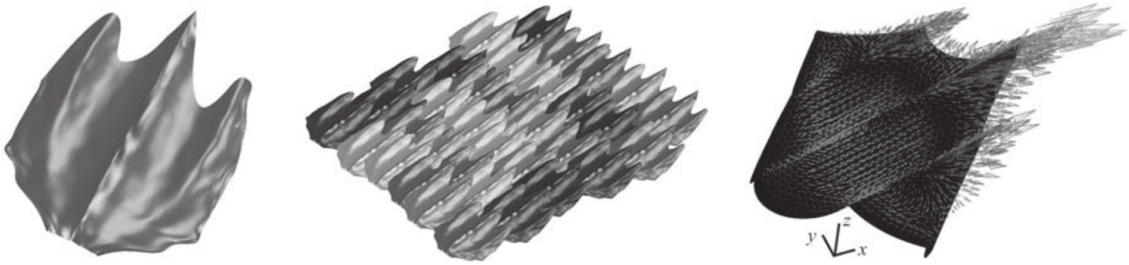


Figure 3.8: Shark skin surface roughness modelling by D.-y. Zhang et al. (2011).

Primarily though, the roughness length and the geometry of the roughness structure both play a part in surface friction. Other work investigates the production of turbulent energy from various roughness structures with the application to improving ship and submarine design (Krogstadt & Antonia, 1999), although it is limited to uniform, small-scale roughness structures.

Studies on mullions (Maruta et al., 1998; Stathopoulos & Zhu, 1990), louvres (Rofail & Kwok, 1999), and balconies (Montazeri & Blocken, 2014; Stathopoulos & Zhu, 1988) indicate significant effects (positive and negative) on pressure in certain surface regions. In all cases the complexity of the effects is remarked upon, making it difficult to generalise the effects of surface detailing on surface pressure. Pressure distributions are influenced by the flow boundary conditions, interference from urban surroundings and the building form (Montazeri & Blocken, 2013). Effects can also be found in more subtle manipulations such as corner chamfering or cutting, and by creating voids, or porous regions, near the edges. Aerodynamic modifications are often made instead of altering the shape of the building: for example, Ilgin & Gunel (2007); Kwok et al. (1986); Tse et al. (2009).

Whilst existing work suggests that surface detailing does have a significant effect on surface pressure, all studies are comparisons of *with detailing* versus *without detailing* for the same overall form. This means that the significance is potentially narrow or over-stated when considering the broader range of form as well. The work by D.-y. Zhang et al. (2011) highlights this issue, as the benefits of the shark skin structure are measured relative to a smooth surface and not including the overall aerodynamic form of the shark. For early-stage building design, the importance of detailing must be relative to the form; and so from this perspective, it has been established that for bluff bodies, pressure drag is generally greater than surface drag (§3.2.1).

3.2.5 Validating simulations

A difficulty with validation, or assessing the accuracy compared to empirical tests, is that CFD is typically used for problems that cannot be easily tested empirically, such as with tall buildings (Addington, 2003). Consequentially, there is no data available in the literature on either surface pressure or flow patterns around existing real tall buildings against which to compare. Considering that every building is markedly different, if validation on one real tall building case was possible the implications on another case would still remain limited. Judging by its absence, the collection of field data for validation is clearly challenging, technically difficult, or costly. In fact, this situation is common for most sophisticated engineering applications, forcing a reliance on validation through similarity analysis or benchmarking.

Boehm (1981) and Blottner (1990) make the distinction between verification as 'solving the equations right' and validation as 'solving the right equations'. That is, the first relates to the CFD solver code development and numerical errors, and the second to its calculation by the user and conceptual modelling errors (Roache, 1997). Although verification is principally out of scope of this thesis, validation is required to establish estimates for both the speed and solution accuracy.

There are generally conservative views towards CFD in the wind engineering literature, with much work invested in validation of simple geometries. Most validation exercises are limited to simple orthogonal geometries like cuboids (the surface-mounted cube is the primary example) and other extruded prisms, as these represent most of the complexities found in bluff bodies. They seek correlation between full-scale, experimental and various numerical results as the debate between using wind tunnels and computational fluid dynamics is extensive.

In the remainder of this section, existing validation studies on FFD and CFD are identified. These give the foundation data for comparative validation or further analysis presented in the following chapter.

3.2.6 Validation: FFD

The FFD solver has been compared with RNG $k-\epsilon$ ($FLUENT$) for four typical indoor flows: i) a fully developed turbulent flow in a plane channel; ii) forced convection flow in a ventilated room; iii) natural convection flow in a tall cavity; iv) and mixed convection flow in a ventilated room (Zuo & Chen, 2009). The intended application is for emergency indoor airflow simulations of smoke transportation, where an intermediate approach between the accuracy, resolution, and time of CFD and multi-zonal simulations is desirable. Additionally, turbulence was accounted for by either adding a constant turbulent viscosity or a zero-equation turbulence model. It was concluded that the FFD was around 50-times faster than the CFD, and that the addition of the turbulence

model actually reduced the comparative accuracy.

In a similar study, the FFD was parallelised to run on a GPU (graphics processing unit) to compare speeds with FFD and CFD on the CPU (Zuo & Chen, 2010). Figure 3.9 shows a linear relationship between computation time and mesh size, with an improvement of at least 10-times compared with the FFD on CPU and 500-times with CFD on CPU.

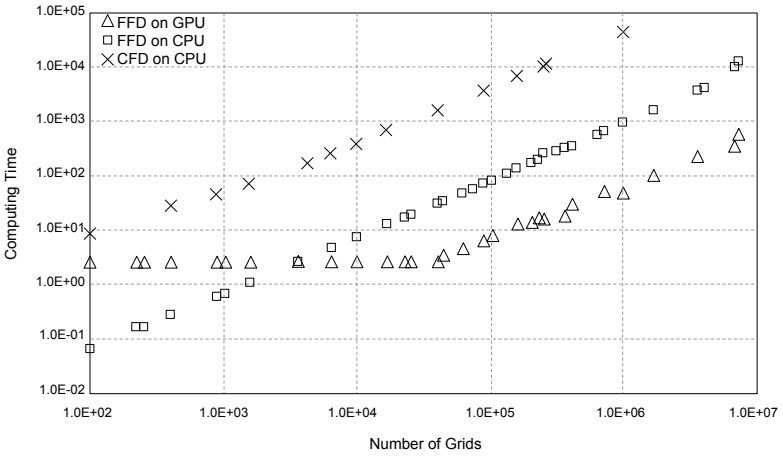


Figure 3.9: FFD and CFD computation time vs. mesh size (Zuo & Chen, 2010).

It was concluded that the FFD could correctly predict the laminar flow (Re=100) present in the test cases. This has been taken to mean that it is fit to use for other flow types, i.e. external wind studies with turbulent flow around bluff bodies. There is a large ambiguity however over its use for anything beyond indoor air flows, even with their concluding statement that it can capture major flow patterns but is not as accurate as CFD. A 3-D version of the same solver was also developed (M. Jin et al., 2012) along with a near-wall treatment, but without a turbulence model it was not tested for fully developed turbulent flows. Again this is supported by Liu et al. (2004) who, with a focal application in computer graphics, state that whilst the FFD is not accurate enough for engineering problems it does *'capture the characteristics of fluid motion'* (p.1).

M. Jin et al. (2013) give the most relevant validation case by simulating natural ventilation in and around buildings, comparing wind-tunnel and CFD measurements with the FFD. The boundary layer wind-tunnel and CFD data used by M. Jin et al. (2013) is originally sourced from Jiang et al. (2003). Jiang et al. applied a power law vertical distribution to the inlet wind profile with the Reynolds number based on the inflow velocity at the building height was $1.4e^5$. The 2x2x1m wind tunnel, based at Cardiff university, used upstream surface roughness to achieve the desired atmospheric boundary layer, and a laser doppler anemometer was used to record one-dimensional velocity measurements. The accuracy of the anemometer is stated as +/- 0.05m/s and the computer-controlled positioning device with +/- 0.5 to +/- 1.0mm error.

In the first study the field velocities around a group of buildings is compared with CFD (*ANSYS Fluent*) and FFD. The velocity contours are shown in Figure 3.10, showing discrepancies especially

downstream or in the wake region of the buildings.

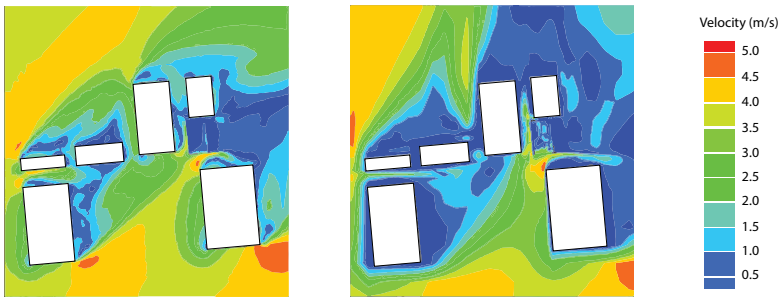


Figure 3.10: RANS (left) vs. FFD (right) comparison of velocity distribution: M. Jin et al. (2013).

In the second study, three cases are modelled for comparison between FFD, CFD, and wind-tunnel: i) single sided, windward ventilation; ii) single sided, leeward ventilation; and iii) cross ventilation. The model geometry and positions for velocity measurements in the stream-wise mid-section are shown in Figure 3.11. The model has two openings of equal size on opposite sides which are alternately open or blocked; the size of the cube was $250mm$ x $250mm$ x $250mm$, with opening dimensions of $84mm$ x $125mm$.

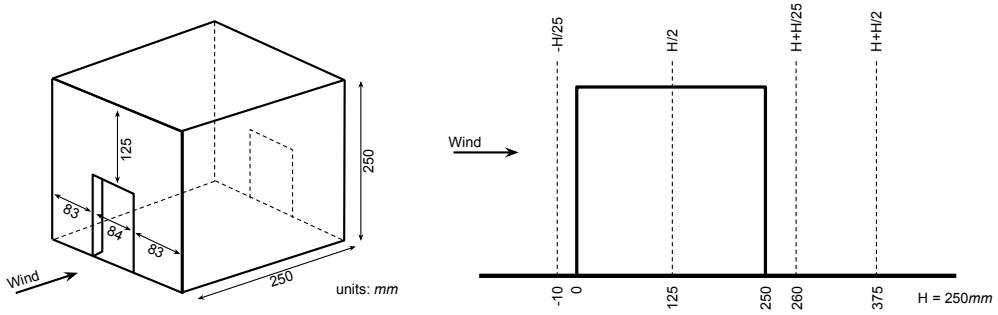
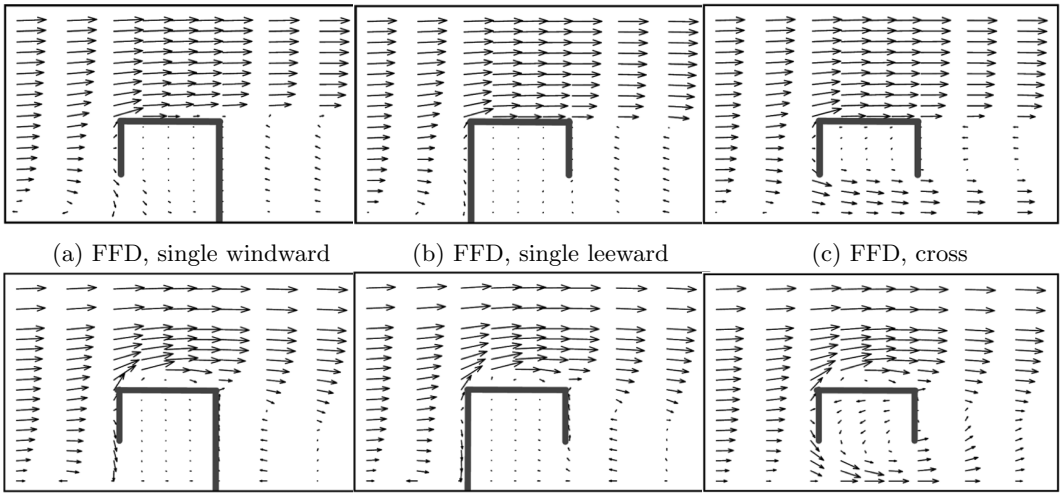


Figure 3.11: FFD validation scale model (left) and evaluation positions (right) (M. Jin et al., 2013).

The field velocity measurements at the stream-wise mid-section plane are shown in Figure 3.12: the FFD on the upper row; the wind-tunnel on the lower row; and the three cases (i, ii, and iii) in each column. It is clear from these images that the FFD under-predicts around the edges of the models, an indication that there is an issue with the turbulence treatment in the flow and particularly close to surfaces.

Figures 3.13 to 3.15 give the field velocity (U/U_{ref}) comparison measurements between FFD, CFD, and wind-tunnel for the three cases: upstream single-sided opening (Figure 3.13); downstream single-sided opening (Figure 3.14); and double-sided opening (Figure 3.15). The four columns are the four vertical lines on the centre stream-wise plane ($-H/25$, $H/2$, $H + H/25$, $H + H/2$) as shown in Figure 3.11.

It was observed that, in each case, the FFD is able to accurately predict the flow field on the windward side , but discrepancies can be found downstream on the leeward side recirculation zone. The greatest discrepancies are found around the top face of the model (Z=0.25m) where flow



(a) FFD, single windward (b) FFD, single leeward (c) FFD, cross

(d) BLWT, single windward (e) BLWT, single leeward (f) BLWT, cross

Figure 3.12: FFD vs. BLWT field comparison at stream-wise mid-section (M. Jin et al., 2013).

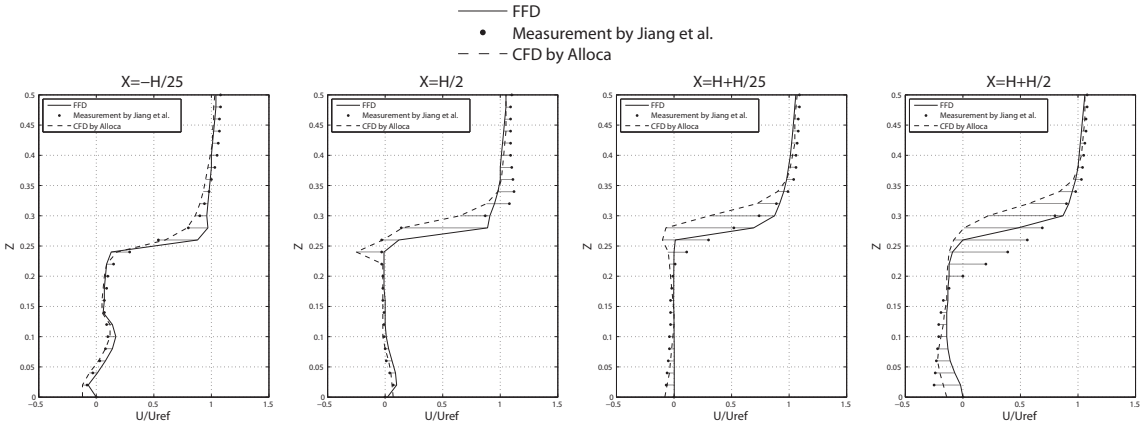


Figure 3.13: FFD vs. WT and CFD field comparison (M. Jin et al., 2013): upstream single-sided.

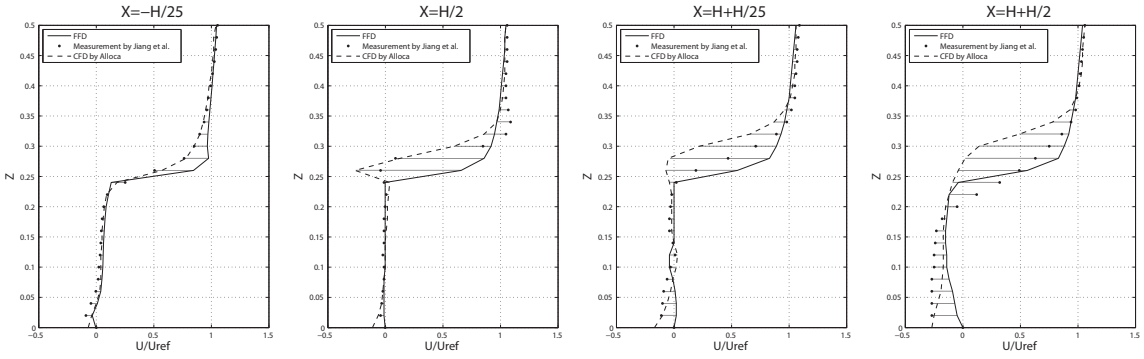


Figure 3.14: FFD vs. WT and CFD field comparison (M. Jin et al., 2013): downstream single.

separation was observed in the wind-tunnel. The flow separation could be captured with a finer mesh however, although this substantially reduces the performance of the FFD.

The primary conclusion from this study, which is in agreement with others, are that the FFD is capable of capturing the major airflow patterns for the demonstrated cases although with a lower

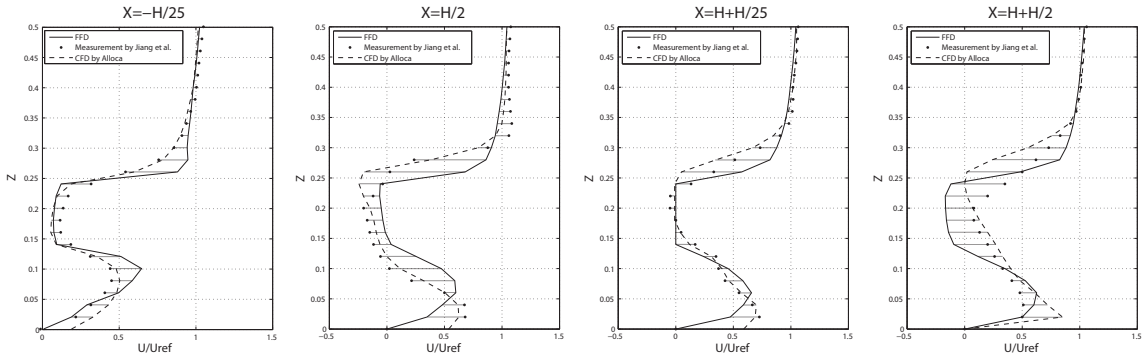


Figure 3.15: FFD vs. WT and CFD field comparison (M. Jin et al., 2013): double-sided.

accuracy than CFD with a turbulence model. The precise errors are calculated from this data in the following chapter (§4.3.5).

3.2.7 Validation: CFD

A recent comparison of studies on a surface-mounted cube by Dagnew et al. (2009) is described in Table 3.2 and Figure 3.16a, covering a variety of simulated, wind-tunnel, and full-scale measurements from various sources. Whilst there are other reviews and individual comparisons in the literature (Delaunay et al., 1995; Gomes et al., 2005; Krajnovic & Davidson, 2002; Murakami et al., 1992; Ochoa & Fueyo, 2004; Rodi, 1997), the comparison by Dagnew et al. (2009) is both the most recent and broad in terms of sources and approaches.

Table 3.2: Summary of surface-mounted cube validation studies, compiled by Dagnew et al. (2009).

Evaluation Method	Source
RANS (k-ϵ)	Bitsuamlak et al. (2008)
RANS (k-ϵ)	Wright & Easom (2003)
RNG (k-ϵ)	Wright & Easom (2003)
RANS (NL)	Wright & Easom (2003)
LES	Lam & To (2006)
Wind-tunnel	Holscher & Niemann (1998)
Wind-tunnel	Richards et al. (2007)
Full-scale	Richards et al. (2007)

Figure 3.16a compares the pressure coefficient along a stream-wise vertical plane on a surface-mounted cube; where position A is the base of the front face and D the base of the rear face. The full-scale measurements by Richards et al. (2007) originate from the Silsoe $6m$ Cube situated in 'open-country'.

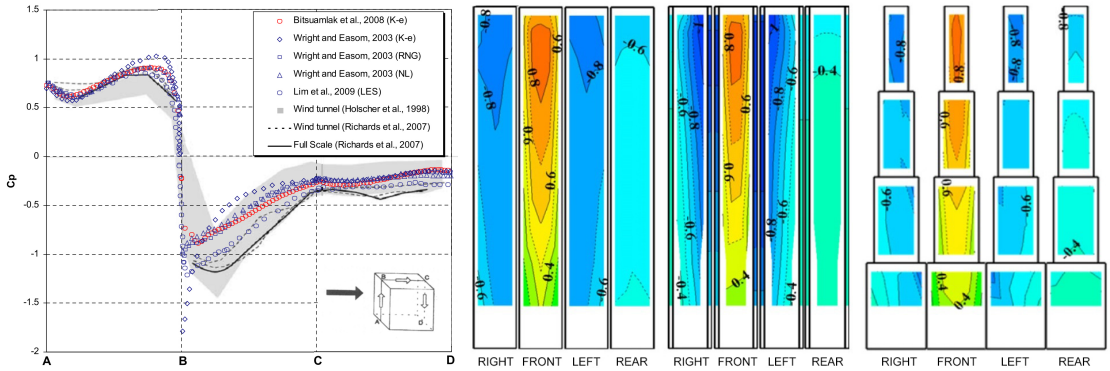
The pressure coefficient is a dimensionless parameter for describing the relative pressure within a flow field. For incompressible or low-speed compressible flows, the relationship is:

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} \tag{3.5}$$

where: p is the pressure at the point of interest $[Pa]$; p_∞ is the free-stream pressure away from

disturbance $[Pa]$; ρ_∞ is the free-stream fluid density $[kg \cdot m^{-3}]$; V_∞ is the free-stream velocity $[m \cdot s^{-1}]$.

LES typically provides the closest results as compared with experimental wind-tunnel and full-scale data. LES data from Lim et al. (2009) and Shah & Ferziger (1997) for the surface-mounted cube will be described and compared directly in the following chapter. The $k - \epsilon$ RANS results also give reasonable results in good agreement with the experimental data, albeit with localised over-prediction around the leading edge (point B in Figure 3.16a).



(a) (b)

Figure 3.16: (a) Comparison of cubic validation studies compiled by Dagnew et al. (2009); (b) Surface pressure from wind-tunnel tests of three tall building models (Tanaka et al., 2012).

For studies besides the surface-mounted cube, there has been no such compilation to date, and so Table 3.3 summarises a selection of individual comparisons. In general, non-cubic validation studies are more fragmented and less comprehensive, tending to focus on a specific project or application.

Table 3.3: Selection of non-cubic validation studies in the literature.

Shape	**Evaluation Method**	**Source**
L- / U-	RNG (k-ϵ) Wind-tunnel	Gomes et al. (2005)
Cylinder	RANS (k-ϵ) Wind-tunnel	Bitsuamlak (2006)
House	RANS (k-ϵ) Wind-tunnel	Delaunay et al. (1995)
Tall building	RANS Wind-tunnel	Colombo et al. (2006)
Tall building	RANS (k-ϵ) Wind-tunnel	Fu et al. (2006)
Tall building	Wind-tunnel	Tanaka et al. (2012)

To attempt to summarise these disparate findings from numerical, wind-tunnel, and full-scale testing on various shaped objects, by comparison with detailed measurements RANS and LES predict well the main features of the complex flows. There is concern on the under-prediction of periodic flows with RANS, which is to be expected given the time-averaging approach. LES does a better job at predicting these turbulence fluctuations and in general gives a more detailed simulation, with the cost of increased computing time. There is the opinion, supported by its

prevalence in industry for predicting general fluid flows, that the inaccuracies inherent in modelling bluff body flows with RANS must be accepted for the meanwhile until LES becomes more practical (Rodi, 1997). It is also strongly noted that, where possible, CFD should be supported by wind-tunnel modelling, so as to use both in a complementary way (Bitsuamlak, 2006).

Generally, in conclusion the surface-mounted cube studies are more amenable to validation since the geometry is simple, easily replicable, and the flow exhibits a range of complex and challenging behaviours (i.e. bluff body flow). In the following chapter, the data identified here will be used for assessing the accuracy and time of the various approaches.

3.3 Machine Learning

Machine learning is proposed as a solution to approximating CFD in order to increase speed whilst retaining simulation accuracy. This approach is generally supported by Augenbroe (2003) who suggests the need for expert knowledge systems to bypass costly simulations. It is cautioned however that artificial intelligence (AI) based approaches have previously been attempted unsuccessfully, limited by their inflexibility or inability to generalise beyond the immediate problem. The suggested reason for this is that as the need for machine learning 'intervention' increases, so too does the problem complexity.

This is however essentially an issue of problem definition; the use of top-down, extrinsic features limits the flexibility to generalise beyond the immediate problem. This thesis, specifically the first hypothesis, addresses this issue by the definition of intrinsic local features which enables greater generalisation.

Related again to the first hypothesis described in the first chapter, i.e. that a larger system can be approximated by local decomposition, Samarasinghe (2007) suggests that learning system behaviour through observational data is key for complex systems with non-linear interactions. Learning such behaviours is often difficult since the systems are typically natural, and therefore can have randomness, heterogeneity, multiple causes and effects, and noise. In fact, even when they are successfully learnt by the computer, they are typically held as intractable computational functions or data structures (decision trees, neuron weights, etc.). That is, predictive models can be constructed from observational data alone, without any a priori knowledge of the system's behaviour. This idea is central to the work of Hanna (2007, 2011) who demonstrates a method using machine learning to approximate the structural behaviour of a small component within a larger system, but far more efficiently than typical simulation. It is demonstrated that approximately optimal solutions can be found by training a machine learning algorithm on examples of other solutions found by a traditional optimisation procedure. The learning described is intended to replace conventional structural optimisation since the method is orders of magnitude faster making it more

amenable to generative design. As motivation, similar to here, it is noted how in optimisation the simulation of system behaviour is often a bottleneck.

The notion of learning is simplified here to mean the construction of general rules, or the extraction of knowledge, from discrete observational data (Wilson, 2010). Human learning uses experiential data and converts it to knowledge, a process that removes highly specific details and generates more generally applicable knowledge (Duffy, 1997). Again, to simplify this inductive process to be used computationally, to transform a set of examples into a model of a system's behaviour the problem must be represented in terms of a vector of input and output features from which the model can be taught to map inputs to outputs (Lu et al., 1991).

The subsequent success of generalisation, or regression, must be measured by its prediction accuracy. There is therefore a compromise between generalisability and specificity when fitting to the data; Figure 3.17 shows (left) under-, (centre) well-, and (right) over-fitting of such data.

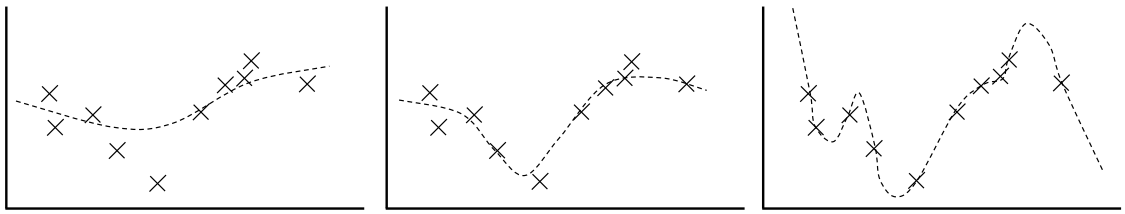


Figure 3.17: Under-, well-, and over-fitting generalisations (Lawrence et al., 1996)

The obvious benefit of machine learning is that the time taken to accumulate prior experiences or observational data and learn from it can be excluded from the prediction time. The whole process can therefore be divided into back-end, offline process time for data collection and learning, and the front-end, online time for prediction. Decomposition of the whole process means that the intensive, time-consuming simulations can be undertaken when time and computational resources are not constrained (Marsland, 2009).

3.3.1 Terminology

The terminology surrounding the problem is diverse indicating a broad array of applications and fields of study. For instance, the term machine learning stems from computer science or artificial intelligence (AI) and is used broadly to convey the concept of the approach. However there are also commonalities with model reduction (reduced-order modelling), meta-, and surrogate modelling from engineering disciplines and are referred to from a systematic or simulation perspective. Regression is also referred to in some cases to convey the notion of extracting continuous functions from discrete data sets of samples as found in statistics. A key difference from statistical regression is the focus on prediction accuracy over understanding; statisticians being typically interested in producing a mathematical description in order to understand trends in the data. And finally,

at the highest level, inductive learning or inference relates to human epistemology and AI (Holland et al., 1987; Hume, 1896; Popper, 1959), providing the original concept for neural network learning. The lack of consistency in terminology is rooted in the immaturity in the specific application of simulation approximation for architectural design and the subsequent need to 'borrow' foundations.

3.3.2 Reduced-order models

Model reduction refers to the creation of approximate representations of system behaviours, namely for computational simulations with slow response times. The aim is fundamentally to create a lower-dimensional system model whilst retaining predictive fidelity (Bui-Thanh et al., 2008; Schilders, 2008). Analyses that are potentially obstructive to the design process may involve partial differential equations (PDEs), such as the Navier-Stokes equations of fluid flow and the Maxwell equations in electromagnetics (Degroote et al., 2010). Reduced-order models (ROMs) are derived from such high-fidelity simulations or models and restrict the output quantities to those of interest (e.g. lift or drag, or a quantity at a single point).

Model order reduction originated in systems and control theory fields where there was a strong need to reduce the complexity of dynamic systems in order to study them. These systems may have of the order of 10^5 to 10^9 equations or variables, and yet their input-output behaviour must be preserved. A commonality of the following studies, each of which is at the core of the model reduction field, is their use of a projection basis for reduction (refer to Schilders (2008) for a comprehensive overview of methods). For instance, principal orthogonal decomposition (POD) is commonly used and is another term for principal component analysis (PCA). This methodology differs from the approach taken in this thesis, but the primary objectives remain the same. Applications tend towards aerospace, i.e. airfoil optimisation (Allaire et al., 2012; Allaire & Willcox, 2013; Lieberman & Willcox, 2012; Ly & Tran, 1999; Robinson et al., 2008; Willcox, 2000), since constrained problem definitions are more amenable to reduction.

3.3.3 Meta-models

Meta-models, surrogate models, and fitness approximation pertain broadly to various engineering disciplines where an approximate representation of a simulation's output is required for analysis or optimisation problems. Such models use approximation techniques to represent the performance space when the cost of evaluating a specific case is obstructively high. Various approaches are surveyed for fitness approximation, specifically for evolutionary optimisation, by Y. Jin (2003) and in metamodelling for engineering design by Simpson, Peplinski, et al. (2001) and Wang & Shan (2007). Again, applications tend towards airfoil optimisation (Meckesheimer et al., 2002; Ong et

al., 2003; Simpson, Mauery, et al., 2001).

3.3.4 Algorithms

The reviews mentioned previously give an array of different available models: polynomials, kriging, artificial neural networks (ANN), and support vector machines (SVM). In addition to these are the random forest (RF) and the projection methods used in reduced-order modelling. Selection of an appropriate algorithm is dependent on the application and type of problem, allowing a basic preliminary filtering of available approaches.

In the case here, a number of decisions are required before pre-selecting the potentially suitable algorithms. Firstly, training data is generated as a single event and so an off-line method is for one-time training; as opposed to a continuously evolving on-line one. Secondly, as the training data has known input and output features the problem is a supervised learning case instead of unsupervised where only inputs are known (e.g. clustering with self-organising maps).

Finally, due to prior knowledge and common sense relating to the complexity of fluid dynamics, it is fairly certain that the relationships being approximated are non-linear. Therefore an algorithm that can model both linear and arbitrarily complex non-linear relationship is required. These three requirements leave the following subset of available algorithms: support vector machines (SVM); random forests (RF); and artificial neural networks (ANN).

Artificial neural network (ANN)

The artificial neural network (ANN) consists of a system of interconnected neurons, typically in at least three layers (input:hidden:output), which can be adjusted to map a set of inputs to outputs (Demuth & Beale, 2002; Haykin, 1999). Each input neuron is connected to every hidden layer neuron, and every hidden layer neuron to the output neuron (response) (Figure 3.18).

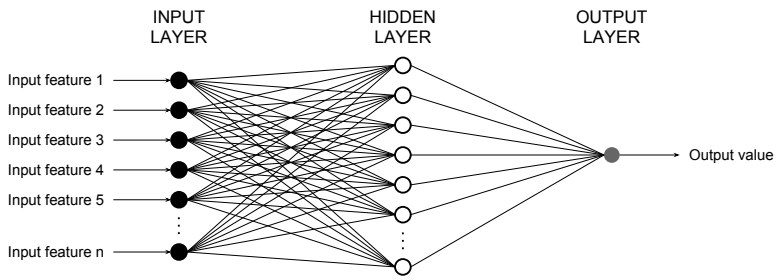


Figure 3.18: Basic neural network topology.

The inputs are initially assigned random weights and summed with a bias to form the hidden layer neuron values. From here, the sum of the weighted inputs and bias form the input to the transfer or activation function to generate the output response. In the case here, of supervised

learning, for adjusting or training the network a set of known input-output samples are used. A back-propagation training method is used (gradient descent) to update the input weights and biases in the direction in which the performance function (the error between prediction and known output responses) decreases most rapidly, i.e. negative gradient. The back-propagation method requires the transfer function to be differentiable, so a hyperbolic tangent sigmoid (tan-sig) transfer function is used (Vogl et al., 1988).

$$tansig(x) = 2/(1 + exp(-2 \cdot x)) - 1 \tag{3.6}$$

Typically in subsequent chapters, the ANN structure $X{:}H{:}Y$ is 22:20:1, i.e. 22 input neurons, 20 hidden layer neurons, and 1 output. The (sigmoid) transfer function places the method in the class of kernel-based learning along with polynomial regression splines (Friedman, 1990; Turner et al., 2007), Gaussian radial basis functions (RBF), and principal components analysis (PCA).

Support vector machines (SVM)

Support vector machines (SVM) are also a kernel-based learning method for classification, function estimation, and density estimation (Cortes & Vapnik, 1995; Vapnik, 2000). The kernel matrix transformation (the 'kernel trick') maps the non-linear input data to a higher dimensional feature space for standard linear classification or regression (Basak et al., 2007; Gunn, 2010). Specific kernel choice depends on each problem; common kernels are linear, polynomial, RBF, and sigmoid. Once transformed, a hyper-plane (linear in the higher dimensional feature space) is fitted to the data to minimise the separation.

A reformulation of the standard support vector machine with a least-squares margin calculation (LS-SVM) as proposed by Suykens & Vandewalle (1999), is used in §5.2 implemented as a toolkit for *Matlab R2012a* (Brabanter et al., 2011; Suykens et al., 2011). A general definition of a least-squares approach is to find a solution that minimises the sum of the squares of the prediction error. In a standard SVM model the hyper-plane optimisation is a convex quadratic programming problem, as compared to the LS-SVM where only linear equations must be solved. Further details of the method can be found in Suykens (2008) and SISTA (2012).

Random forest (RF)

Breiman (2001) formulated the initial random forest development; a learning algorithm consisting of a collection of decision trees where each individual tree casts a unit vote for the most popular output (ensemble learning). The random forest is implemented in *Matlab R2012a* as the TreeBagger algorithm (Matlab, n.d.). A recent review of the literature surrounding random forests and a

development of its applicability is given by Criminisi et al. (2011).

The random forest is used in §5.3.5 for calculating the relative significance of each input feature to the output, and in §5.3.7 for comparison with the generalisation ability of the ANN.

ANN vs. SVM vs. RF

The primary machine learning algorithm used in this thesis is the artificial neural network; however, the random forest and support vector machine have been used in isolated instances. It was seen that the ANN and RF performed significantly faster than the SVM (§5.3.1), therefore making them more suitable to the large data sets involved in the local feature definition. The RF tended to over-train in comparison to the ANN which generalised better (§5.3.7).

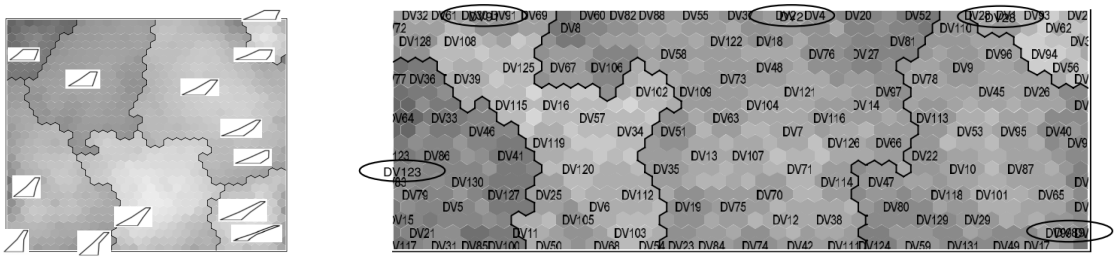
3.3.5 Applications: CFD approximation

A range of CFD methods were presented in the previous section, each of which can be considered as a different approach to approximating fluid flow. These methods can be described as solver approximations, in that they directly relate to the the underlying physical system, whereas solution approximations as described here are of the simulation itself.

It is usually necessary to constrain a problem in order to make an approximation. This can be achieved by using low-order numerical schemes, studying only 2-D flows, using orthogonal or simplified geometry, assuming low Reynold's numbers etc. All of the constraints act to simplify the problem to make it tractable, but in doing so the complexities that make it a real, and interesting, problem are largely removed.

Another application for approximating CFD is the extraction of meaningful performance knowledge. For example, with multi-objective optimisation it can become difficult to visualise more than three objectives of a Pareto set; however a self-organising map (SOM) can be used to reduce or project the higher dimensionality data down to a 2-D space. The SOM is an unsupervised artificial neural network (ANN) used for data mining. By reducing the dimensionality, the data can be clustered in a 2-D representation, keeping the essential topology intact, and making it substantially more understandable for a human. In Figure 3.19 this has been used to cluster wing planform shapes by their objective function values (Chiba et al., 2005; Obayashi & Sasaki, 2003). Note that the two axes do not correspond directly with any variables or objectives, rather they are the 2-D space of the ANN output layer.

Applying learning to this kind of knowledge extraction is useful for making complex problems more comprehensible, however its aim is not to directly help with making new predictions but rather to understand what has already occurred.



(a) (b)

Figure 3.19: Self-organising maps: (a) objective function and typical wing planform shapes; (b) cluster-averaged design variables (Obayashi & Sasaki, 2003).

3.3.6 Applications: interference approximation

Interference has been described earlier (§3.2.2) and a selection of typical studies given in Table 3.1. In an urban context this has significant impact on the performance of the building in question, by increasing or decreasing the forces beyond what they would be in an open setting. There have been a number of studies analysing the effects that a small number of adjacent structures have, leading to the development of the Interference Factor (IF) as a ratio between the surface pressure with and without interference from adjacent structures.

Table 3.4: Summary of existing interference global parameter generalisation studies.

No.	Evaluation / Regression method	O.	SD.	AR.	C.	α	Source
2	WT / Polynomial		$\bullet_{X,Y}$	•		0.14	Khanduri (1997)
2	WT / RBF		$\bullet_{X}$	•		•	English & Fricke (1999)
2	WT / RBF		$\bullet_{X,Y}$				Khanduri et al. (1997)
2	WT / RBF	•	$\bullet_{X,Y}$	•		•	Khanduri (1997)
2	WT / RBF		•	•		•	A. Zhang & Zhang (2004)

No. Number of Study Buildings; **O.** Orientation; **SD.** Separation Distance; **AR.** Aspect Ratio; **C.** Configuration; α Wind profile exponent; X is direction horizontally perpendicular to flow; Y stream-wise; and Z vertical.

The typical formulation of interference studies in the literature is amenable to machine learning in that there are a finite set of input (spatial configuration) and output (IF) parameters. In three cases, a radial basis function (RBF) artificial neural network (ANN) is used with wind-tunnel data (English & Fricke, 1999; Khanduri et al., 1997; A. Zhang & Zhang, 2004) with the objective to create generalisations about spatial configurations of tall buildings.

However, there are clear deficiencies in this approach that will be addressed in later chapters: firstly, all of the studies use similar cuboids since form is not considered; secondly, basic configurations of typically only two or three buildings are included due to the combinatorial explosion; and thirdly, a lack of output data since the IF can only provide a factor indicating the change over the whole model.

3.3.7 Shape features

A component of the methodology involves a localised description of a sample point or vertex, constituting the input vector for the machine learning. These are predominantly shape or topological characteristics, but also a description of the local fluid flow. With discretised surface representations (meshes) there is often a need to describe local shape features.

An exemplar case is for scale-invariant surface descriptors for the matching of molecular surface regions to identify potential chemical functionality, i.e. binding molecules often have locally complementary shapes (Cipriano et al., 2009; Feng et al., 2013). When calculating surface curvature, the distance or neighbourhood size must be included, shown in Figure 3.20. Left to right the features are: minimum curvature; maximum curvature; mean curvature; and Gaussian curvature.

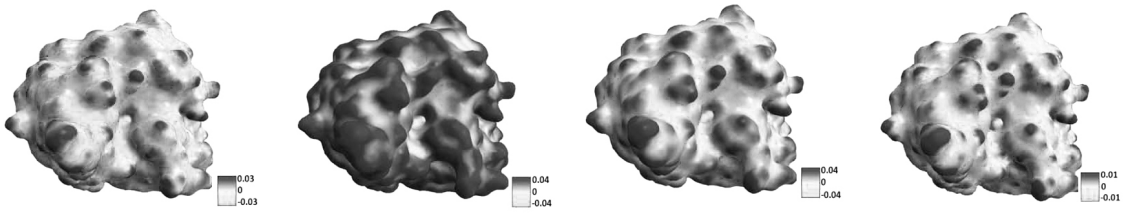


Figure 3.20: Shape analysis for protein shape matching in biochemistry (Feng et al., 2013).

There are a large number of similar studies on mesh curvature, edge detection, and invariant shape analysis, for example (Darom & Keller, 2012; Dong & Wang, 2005; Hubeli & Gross, 2001; Hubeli et al., 2000; Jiao & Heath, 2002; Szilvasi-Nagy, 2006; Walter et al., 2008). Applications can range from chemistry, to rationalising and reconstructing 3-D scanned models, to identifying constant features in images for camera stabilisation.

CFD traditionally originated in aeronautics and astronautics, as such there is a large quantity of work directed towards shape optimisation of airfoils, fuselages, and turbine blades. Often a conventional optimisation routine will generate large sets of simulation data, from which knowledge can be extracted for subsequent problems. In one case, such a set of 200 models of turbine blades is used with a decision tree to create a relationship of point deformation between models and their change in surface pressure (Graening et al., 2008). This means that areas of high sensitivity can be mapped onto a base geometry (which has been decided beforehand) and used to focus subsequent analyses. This work is extended further to be incorporated with an evolutionary optimisation process, so as to use the information extracted from previous cases (the surrogate model giving sensitive regions) to create non-random initial populations of solutions and to guide the evolution (Ramanathan & Graening, 2009).

Whilst Graening et al. do not learn the function between local shape features and pressure to make predictions on new cases, the ability, however, to generate the sensitivity at a point from

its deformation is an inspiration for two key elements of this thesis. Firstly, it highlights the importance of showing the pressure distribution over the entire model rather than calculating a global metric of a design's success, i.e. considering the problem locally rather than globally. This will be shown in chapter 5 where a top-down model approach is used to predict a single peak surface pressure for the entire model, which turns out to be of limited value to the designer. Secondly, the use of top-down models (those found in parametric models with global variables) have an inherently limited flexibility. They can be adjusted infinitely within the bounds of the logic of the model parameters, however it is firstly difficult to alter these foundations at later stages and secondly each model will have a different logic, variables, or set of dependencies. It is therefore difficult to use these global variables (for example, a parametric tall building may have *Height* and *Taper Factor* as two of its defining variables) as input features to learning. If they are used, the problem being learnt is restricted to that logic. Graening et al., on the other hand, use a local mesh vertex deformation as input feature. Since all CFD simulations require a surface mesh of the geometry to be generated, it is a relatively simple generalisation to use the mesh data and its derivates as input for the learning.

Further generalisation of the method is proposed, specifically for automobile design, in particular for the detection of design novelty or for characterising families of similar products (Graening & Sendhoff, 2014). The key similarity with this thesis is the use of unstructured surface meshes as the basis geometric representation, which in itself is a good foundation for generating training data due to the high acceptance in design practice. The distinction however is in the definition of the actual learning process and feature vector: Graening & Sendhoff (2014) use a deformation metric from a base case; as opposed to the here proposed broader shape description.

Their proposed shape mining process (Figure 3.21) focuses on the extraction of performance data from conventional analysis processes for compilation of a large database. From which a meta-representation, or reduced-order model, can be created and used for sensitivity analysis, concept retrieval, and interaction analysis. These data modelling and knowledge formation components link back holistically to knowledge utilisation and decision making processes (DMPs).

Rendall & Allen (2008) use spatial and behavioural parameters as input features to a radial basis function (RBF) for interpolating and merging computationally and experimentally derived pressure coefficient (lift and drag) values for airfoil analysis. They use an input feature vector, $\mathbf{X}$, relating to the pressure coefficient output, Y:

$$\mathbf{X}\{x, y, z, a, M, Re\} \rightarrow Y\{C_p\} \tag{3.7}$$

consisting of the x, y, z spatial position, the angle of attack a, the Mach number M, an the Reynolds number Re. Whilst their method proved successful for interpolation between behavioural

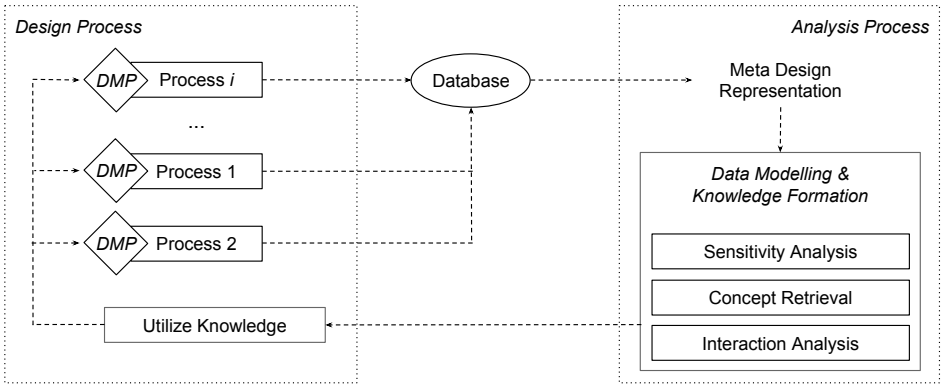


Figure 3.21: High level view on the shape mining framework (Graening & Sendhoff, 2014).

characteristics (a, M, and Re) and data sources (CFD and wind-tunnel), it is limited to a single geometry, thus the use of explicit spatial positions (x, y, and z). For cases of multiple geometry, spatial positions become non-unique between geometries and can therefore not be used within the feature vector.

Using spatial positions (or mesh node numbers) for feature vector is also proposed by Srinivasan & Malkawi (2004). In this case, an ANN is used to predict post-processed CFD data for rapid visualisation and interpolation of boundary conditions with an augmented reality user display system. The input, $\mathbf{X}$, and output, $\mathbf{Y}$, feature vectors are defined as:

$$\mathbf{X}\{n, S, P\} \rightarrow \mathbf{Y}\{T, V\} \tag{3.8}$$

where T is the air temperature and V is the air speed at a node, n is the mesh node number (1224 nodes in the cubic room), S is the supply temperature, and P is the supply pressure. Again the proposal is limited to a single geometry by the use of spatial positions or node numbers corresponding consistently with fixed locations. The limitation again is that for multiple or varying geometry the positions are non-unique.

3.3.8 Fluid features

An extension of the local shape features described so far to take into account urban wind interference is the addition of local field properties (wind speed). This methodology is described in §5.4 and §6.3. This is a novel approach for both interference and in computational wind engineering, however there are some precedents in general CFD data mining.

Existing work on extracting features from the fluid field is primarily focused on knowledge extraction or data mining from large sets. Post et al. (2002) review a number of applications such as flow topology classification, vortex identification and tracking, shock wave detection, and separation/attachment detection. Identification of flow characteristics (vortex cores, sepa-

ration/attachment, and shock waves) can be 'mined' during a CFD simulation (Gosnell et al., 2012). In extremely high resolution problems (up to 300 million grid-point transonic turbofan simulation with RANS) it can often take weeks or months for a single run to converge on a stable result. Therefore employing feature detection during the simulation can give insight into the development and stability of the actual fluid structure rather than physical properties of the simulation (e.g. mass and momentum residuals). This results in considerable time-savings if features of interest can be observed and tracked directly, potentially allowing the user to cut short the simulation when they are confident that the flow is stable enough for the accuracy requirements of the problem.

Weinkauf et al. (2012) focus on the characterisation of streamline curves from flow fields and their mathematical rationalisation to a common form. This rationalisation could potentially be used for expressing the entire flow field in a manner amenable to reduced-order modelling techniques, i.e. geometry definition as input feature and flow streamline curve set as output. Whilst this is an attractive idea, a number of problems are presented; primarily that in turbulent bluff body flows the streamlines are often far more complex than in laminar flows. And secondarily, unless a direct relationship is established between local geometry points and streamlines, the reduction will be global in nature, relying on global parameter descriptions.

3.4 Review Summary

3.4.1 Trends

In the review it is possible to find the general directions of the various fields that have been covered which are summarised below:

- Tall buildings are increasing in quantity, height, and geometric complexity, enabled by computational developments in construction, fabrication, coordination, design, and analysis. Increased use of computational design and analysis, although integrated to some degree, are still distinct for intensive analysis, such as CFD, within generative design.
- Generally, there is a shift towards the use of CFD at more stages throughout the design process, with physical wind-tunnel tests usually reserved for final validation. This is primarily a question of confidence in simulation, but there are many other factors such as cost, ease of use, and output.
- There is a considerable effort towards integrating a variety of building performance data, optimisation tools, and parametric software for early stage generative design. This is epitomised through *Bentley's* development of a cloud analysis framework which aims to consolidate disparate components of the process.

- There is also a strong interest in using CFD for early-stage design guidance rather than later project stage analysis, evident in the rising application of low-order FFD solvers. Solver development is focused on either speed or accuracy, observable in FFD and CFD validation and speed assessments.
- As well as the rise in simulation, the adoption of machine learning in nearly all aspects of engineering appears now to be filtering through to architectural design. This trend for engineering practice or computer science to be simplified and transposed to design is a recurring process (e.g. CAD, parametric software, rendering, and simulation).

3.4.2 Opportunities

The following areas have been identified as being open to further study:

- There has been a very limited amount of existing work generally on integrating machine learning with CFD. Existing research focuses on: i) knowledge extraction from large data sets, after-the-fact, rather than for improving evaluation speed; ii) top-down problem definitions, such as for urban wind interference generalisation; iii) or on boundary condition interpolation with constant geometry or spatial positions.
- Work on development of CFD codes is primarily focused on accuracy rather than identifying appropriate speed-accuracy compromises. Faster solvers have been developed, but their accuracy is tailored so as to allow real-time interaction for computer graphics applications. There is therefore an opportunity to explore, through applying machine learning, an approach which is fast-yet-accurate.
- Finally, the field of urban wind interference is currently limited to combinatorial global parameter sensitivity analyses. There are only a few cases generalising a small numbers of identical prismatic cuboid geometries to extract simplistic rules-of-thumb. Alternative approaches must be proposed which address the constraints of the existing literature.

Chapter 4

Fluid Simulation

In this chapter, the computational fluid dynamics (CFD) simulation methodology will be described, along with comparative validation of a fast fluid dynamics (FFD) solver and the primary CFD tool, *CFX* 13.0 (ANSYS, 2014) which is used in subsequent chapters. The primary objective of this chapter is to establish the accuracy of FFD, RANS, LES, wind-tunnel, and full-scale data relative to one another. This information is combined with further errors associated with the machine learning approximation described in the following chapter to give an error relative to the 'ground-truth', i.e. the total errors of the reduced-order model (ROM) established in Chapter 7 is the basis CFD (RANS or LES) error plus the ROM error.

Table 4.1: Summary of Chapter 4 studies.

	Study	Section
CFX RANS validation		
	Mesh sensitivity, cube	4.2.3
	RANS vs. literature, cube quantitative	4.2.4
	RANS vs. literature, qualitative	4.2.5
FFD validation		
	FFD vs. RANS, single-variate	4.3.3
	FFD vs. RANS, super-ellipse	4.3.4
	FFD vs. WT and RANS, field	4.3.5

4.1 Accuracy Definition

Simulations can be generally defined as representations of certain aspects of a system's real-world behaviour. Typically it is not possible to model every aspect of the full system, therefore simulations tend to focus on accurately conveying limited properties of interest (even more so for reduced-order models). The constrained definition of accuracy adopted henceforth deals solely with surface pressure, and with a 'ground-truth' being established between approximations (computational: FFD, RANS, LES. experimental: WT) and reality (full-scale measurements). Note that the terms 'accuracy' and 'error' are used inversely, i.e. 20% error = 80% accuracy.

A hierarchy must be established with which to compare the data, in that the accuracy of each study is typically relative to another (Figure 4.1). For instance, the accuracy of RANS may be assessed

relative to the wind-tunnel, and the reduced-order model studies relative to RANS. The hierarchy of accuracy, in the approaches considered, is therefore as follows (from highest to lowest): Full-scale (Full); Wind-tunnel (WT); Large Eddy Simulation (LES); Reynolds-Averaged Navier-Stokes (RANS); Fast Fluid Dynamics (FFD). Where possible, the accuracy of an approach is measured against the most accurate available source. Subsequently, the reduced-order model data is itself measured against RANS or LES, from which it is derived.

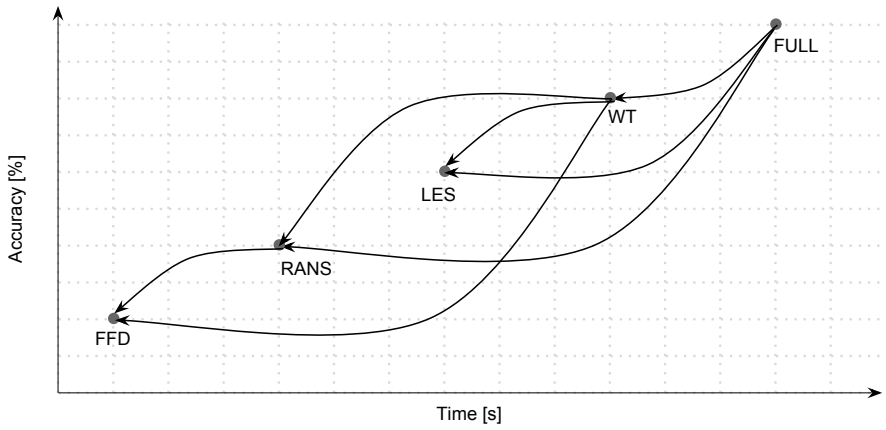


Figure 4.1: Notional hierarchy of derived accuracy.

4.2 Computational Fluid Dynamics (CFD) Validation

The primary CFD software being used henceforth in this chapter is *ANSYS CFX 13.0*; a steady-state Reynolds-Averaged Navier-Stokes (RANS) approach with a $k-\epsilon$ turbulence model (§3.2.3). For subsequent chapters, where *CFX* is also used for large eddy simulation (LES), additional simulation parameters are given.

4.2.1 Simulation methodology

The simulation methodology for the validation cases is as follows:

i) *Geometry:* The model geometry is created in *GC*, although any CAD software can be substituted. The benefit of *GC* is that a parametric model can be created to quickly generate multiple variations. The geometry, which at this stage consists of a single 'solid' body with the focal geometry boolean differenced from the domain, is exported in the *.*IGES* format. Solids are used instead of b-spline surfaces or surface meshes at this stage to ensure that the model is 'air-tight' and has no holes or gaps.

ii) *Meshing:* The rest of the process is completed using *ANSYS Workbench*, where all the stages of the simulation can be brought together and potentially automated. The **.IGES* file is imported and meshed according to the general parameters given in Table 4.2. These may

be varied slightly depending on the model complexity and available computational resources. The resulting volumetric and surface mesh constitutes the spatial discretisation (§3.2.3).

iii) *Boundary Conditions:* Defining the simulation boundary conditions means assigning physical properties to the inert geometry. The fluid properties, surface characteristics, inlet and outlet flow conditions, and solver control parameters are all assigned in *CFX Pre*. The general parameters used throughout, unless otherwise stated, are given in Table 4.3. A **.DEF* definition file is produced which holds all of the data necessary to run the simulation.

iv) *Simulation:* The definition file (or files, batch simulations can be run where necessary) is then passed to *CFX Solve*. The simulation time is directly related to the number of nodes or vertices (i.e. model complexity) in the mesh (Figure 4.3), as well as the number of iterations required to reach convergence.

v) *Data Extraction:* Once the simulation is complete, the **.RES* result file can be viewed and post-processed in *CFX Post*. Relevant data throughout the domain or surface mesh can be extracted for further analysis.

Table 4.2: *CFX* RANS meshing parameters.

Parameter	**Value / description**	**Units**
Boundary surfaces		
Element size	5	m
Model surfaces		
Curvature normal angle	18	$^\circ$
Minimum size	0.20	m
Maximum face size	0.25	m
Maximum size	0.25	m
Growth rate	1.4	-
Inflation transition ratio	0.77	-
Inflation maximum layers	3	-
Inflation growth rate	1.2	-
Domain		
Method	Unstructured (tetrahedrons)	-
Algorithm	Patch independent	-
Maximum element size	0.25	m

Table 4.3: *CFX* RANS setup parameters and simulation boundary conditions.

Parameter	**Value / description**	**Units**
Fluid properties: air		
Temperature	25	$^\circ C$
Density	1.185	$kg \cdot m^{-3}$
Reference pressure	1.0	atm [a]
Dynamic viscosity [g]	$1.831e^{-5}$	$kg \cdot m^{-1} \cdot s^{-1}$
Boundary conditions - Parallel walls		
Mass and momentum	Free-slip [e]	-
Boundary conditions - Ground		
Mass and momentum	No-slip [d]	-
Wall roughness	Smooth [h]	-
Boundary conditions - Model		
Mass and momentum	No-slip [d]	-
Wall roughness	Smooth [h]	-
Boundary conditions - Inlet		
Flow regime	Subsonic	-
Turbulence	Medium intensity and eddy viscosity ratio [c]	-

(...continued on next page)

CFX RANS setup parameters and simulation boundary conditions (Table continued.)

Boundary conditions - Outlet		
Flow regime	Subsonic	-
Flow direction	Normal to boundary	-
Relative pressure	0.0	Pa [b]
Turbulence	Medium intensity and eddy viscosity ratio [c]	-
Inlet wind profile (§4.2.2)		
Inlet velocity	$U_x = U_r.(Z_x/Z_r)^{\alpha}$	$m \cdot s^{-1}$
Reference velocity, U_r	10	$m \cdot s^{-1}$
Height, Z_x	(Varies)	m
Reference height, Z_r	10	m
Profile coefficient, α	0.143	-
Domain properties		
Simulation type	Steady-state	-
Buoyancy model	Non buoyant	-
Domain motion	Stationary	-
Reference pressure	1.0	atm [a]
Turbulence model	k-ϵ	-
Solver control		
Turbulence numerics	High resolution	-
Convergence residual target	$1e^{-6}$	-
Convergence residual type	RMS	-

Notes for Table 4.3:
[a] Reference pressure = 1 atm, standard atmosphere, equal to 101325 Pa;
[b] 1 Pa, Pascal, is equal to 1 $N \cdot m^{-2}$;
[c] The eddy viscosity ratio is the ratio between the turbulent viscosity and the molecular dynamic viscosity;
[d] No-slip boundary condition means that at a solid boundary the fluid velocity will be zero, i.e. U_{Wall}=0;
[e] Free-slip boundary condition has a velocity component parallel to the wall, but with velocity normal to the wall and the wall shear stress set to zero, i.e. $U_{n,Wall} = 0$ and $\tau_w = 0$;
[f] The residuals are calculated as the imbalance in the linearised system of discrete equations;
[g] The dynamic viscosity is the tangential force per unit area required to move one horizontal plane with respect to the other at unit velocity when maintained a unit distance apart by the fluid.
[h] A 'smooth' wall roughness was applied to the ground and the model in the absence of other details in order to maintain the simplicity and stability of the simulation.
All technical derivations and mathematical definitions are contained in the software manual (ANSYS, 2009).

4.2.2 Methodology: inlet wind profile

In section 3.2.1 the vertical power-law wind profile distribution was explained. This is defined by the following equation (Figure 4.2):

$$U_x = U_r. \left(\frac{Z_x}{Z_r} \right)^{\alpha} \tag{4.1}$$

with an exponent α of 0.143 to represent neutral stability. Air stability refers to the amount of turbulence in the ambient atmosphere, and can be categorised by the Pasquill stability classes (*A: Very Unstable*, *D: Neutral*, *F: Stable*). These classes basically relate to the propensity of the air to change its current condition, i.e. where a *Very Unstable* condition is sensitive to change and a *Stable* condition is resistant to change.

The power-law profile is one of the simplest available and is used because in practice, without specific information, it is best to limit the number of assumptions in order to remain general. That is, instead of including a more complex wind profile at this stage, when interference is present then the addition of surrounding conditions can be modelled explicitly (§5.4 and §6.3).

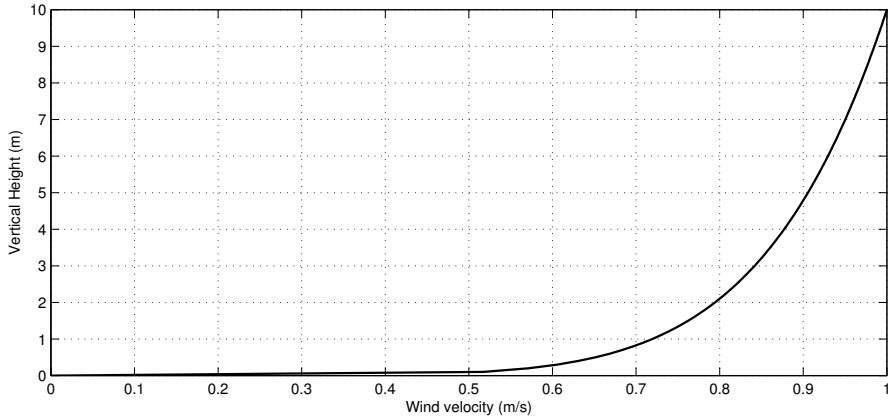


Figure 4.2: Power law vertical wind profile with neutral stability conditions.

4.2.3 Methodology: mesh sensitivity analysis

Simulation results should ideally be independent of the method used to obtain it; with CFD this is particularly important for the approximation involved in spatial discretisation, or meshing. Finer mesh resolutions directly increase computational time; a relationship shown in Figure 4.3, where CPU time per solver iteration is plotted against the number of nodes in the mesh. The linear trend has an average of 0.0001 $seconds \cdot step^{-1} \cdot node^{-1}$ at R^2=0.97. The 40 meshes used here were taken at random from the procedural model training set in §6.2.

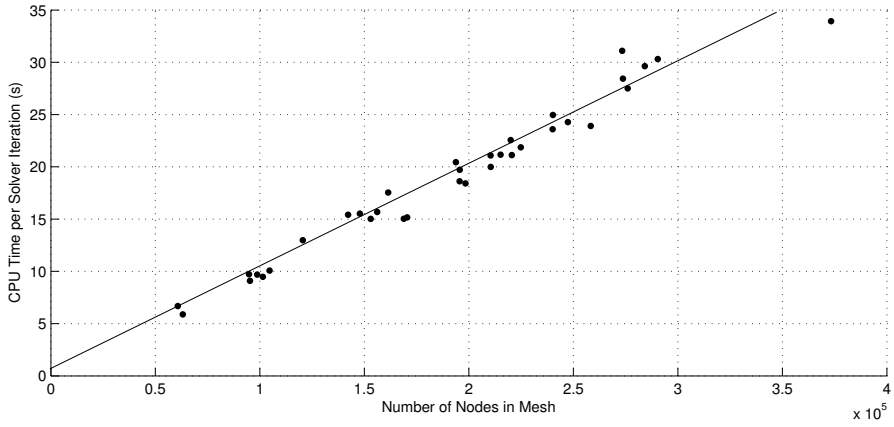


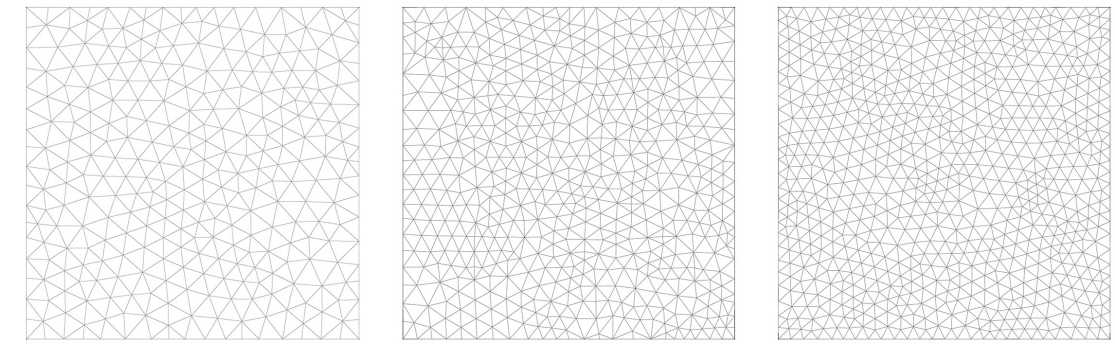
Figure 4.3: *CFX* performance for a random selection of simulations.

To specifically test the importance of mesh resolution on the sensitivity of the solution in *CFX* RANS, the surface-mounted cube, with edge length $10m$, is simulated with various mesh element sizes (Table 4.4). The *element face / edge size* defines the target length of a single tetrahedral element edge; *nodes*, **N**, and *elements*, **E**, are the number of tetrahedral vertices and tetrahedrons themselves; *domain* and *model* are the free fluid volume and the focal model surface respectively. As noted, there is a linear trend between the number of elements or nodes and the total solver CPU time (Table 4.4);.

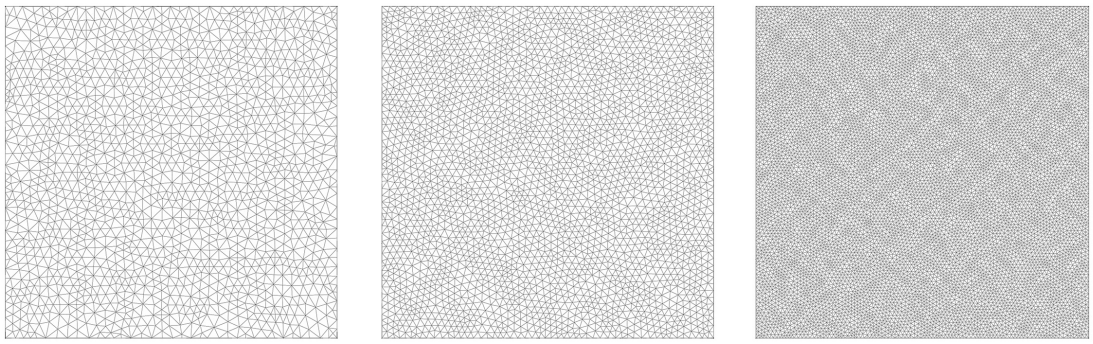
Table 4.4: Mesh sensitivity: parameters and metrics.

Element face / edge size $[m]$	$\mathbf{N}_{domain}$	$\mathbf{E}_{domain}$	$\mathbf{N}_{model}$	$\mathbf{E}_{model}$	**Total CPU time, t** $[s]$
0.60	20824	112581	1826	3290	396
0.50	31358	171534	2463	4503	676
0.40	53121	291643	4320	8084	1206
0.30	65729	361861	6642	12559	1366
0.20	137715	758483	13864	26678	2821
0.10	463361	2547617	53631	105138	9515
$f(t)$	$48.85t$+1408	$268.67t$+8263	$5.77t$-1588	$11.36t$+3547	
R^2	0.9998	0.9998	0.9983	0.9982	

The mesh resolution difference are shown in Figure 4.4, ranging from the coarsest on the left to the finest on the right. The simulation time over this ranges from $396s$ ($6.6mins$) to $9515s$ ($158.6mins$).



(a) $0.6m$ (b) $0.5m$ (c) $0.4m$

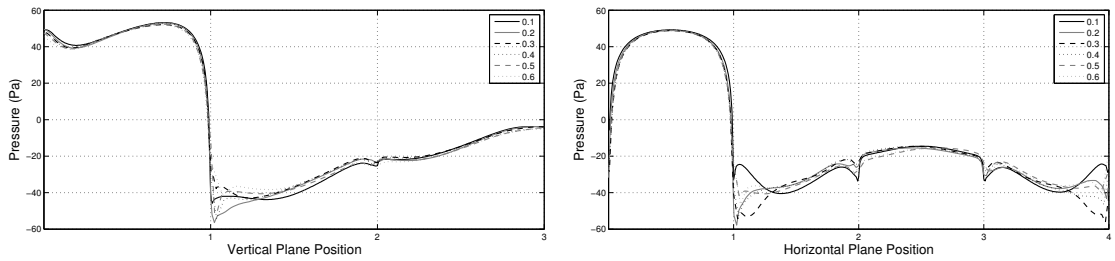


(d) $0.3m$ (e) $0.2m$ (f) $0.1m$

Figure 4.4: Cube mesh top faces with varying element sizing.

To compare the difference between mesh resolutions, the pressure was extracted at the intersection of a horizontal and a vertical plane cutting through the cube (Figure 4.6a). The greatest differences are around the front face edges: after point 1 on the vertical plane is the top face just after the front edge; and on the horizontal plane, after point 1 and before point 4 are on the side faces immediately after the front edges. These three edges are where flow separation occurs, i.e. turbulence is generated.

The pressure variation between increasingly fine mesh resolutions at these localised regions is 14.17% (near point 1, difference between $0.20m$ and $0.30m$) on the vertical plane and 31.17%



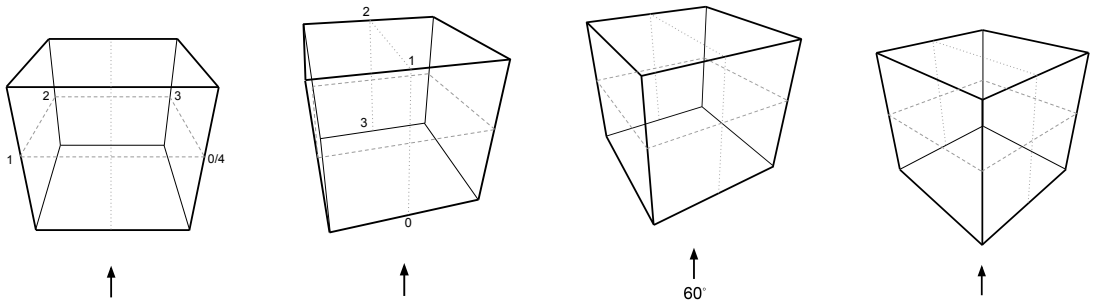
(a) Vertical plane (b) Horizontal plane

Figure 4.5: Mesh sensitivity of cube at various resolutions.

(near point 1, difference between $0.10m$ and $0.20m$) on the horizontal. The conclusion from this sensitivity study is generally that *CFX* is relatively tolerant to varying the mesh size. Sub-grid turbulence models are applied effectively which can compensate for mesh sizes which, so long as they adequately represent geometric features, can be much larger than the turbulence dissipation scale. Whilst there are localised errors around regions of separation, the global pressure distribution remains accurate.

4.2.4 Results: quantitative validation

Since the significant majority of validation studies in the literature (i.e. those identified in §3.2.7) focus on surface-mounted cubes, the following validation comparisons are with existing full-scale, wind-tunnel and CFD data. A cube was oriented at 15° increments from 90° (leading face perpendicular to the flow direction) through to 45° (edge leading to the flow direction), as shown in Figure 4.6.



(a) 90° (b) 75° (c) 60° (d) 45°

Figure 4.6: Cube orientations relative to wind direction.

A vertical and horizontal plane was placed through each, oriented with the cube, and the pressure coefficients calculated along the resulting cube-plane intersection line. The pressure coefficient is a dimensionless parameter for describing the relative pressure within a flow field (§3.2.7). For

incompressible or low-speed compressible flows, the relationship is:

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} \tag{4.2}$$

where: p is the pressure at the point of interest $[Pa]$; p_∞ is the free-stream pressure away from disturbance $[Pa]$; ρ_∞ is the free-stream fluid density $[kg \cdot m^{-3}]$; V_∞ is the free-stream velocity $[m \cdot s^{-1}]$.

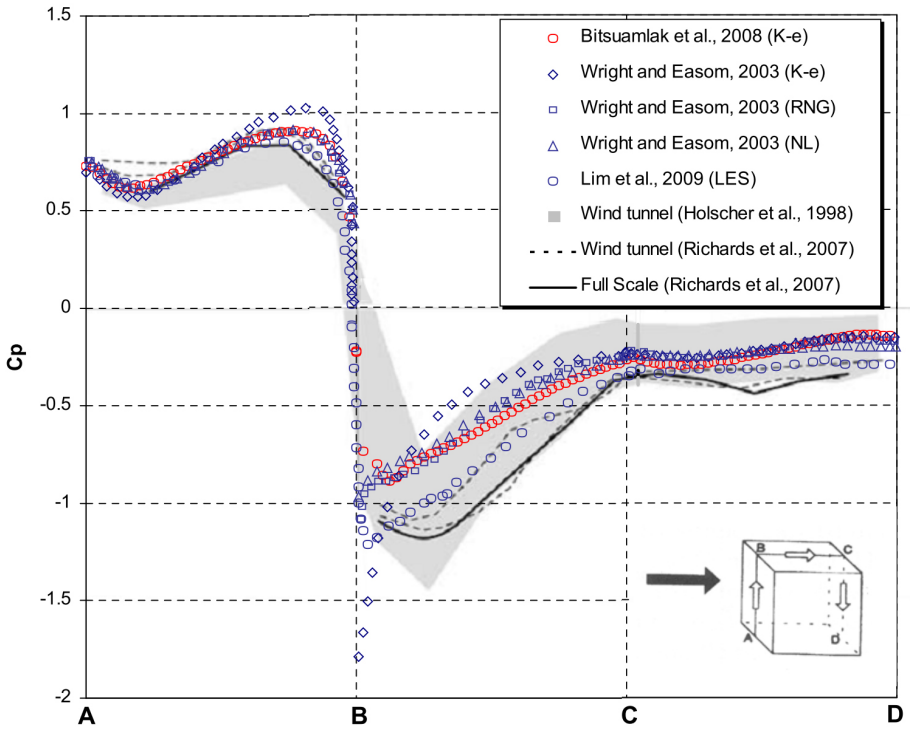
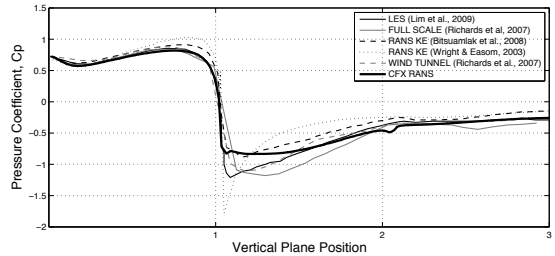


Figure 4.7: Comparison of cubic validation studies compiled by Dagnew et al. (2009).

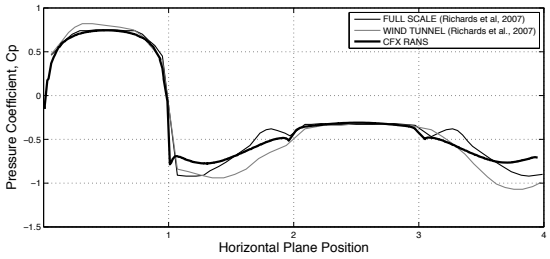
LES typically provides the closest results as compared with experimental wind-tunnel and full-scale data. LES data from Lim et al. (2009) and Shah & Ferziger (1997) for the surface-mounted cube will be described and compared directly in the following chapter. The $k - \epsilon$ RANS results also give reasonable results in good agreement with the experimental data, albeit with localised over-prediction around the leading edge (point B in Figure 4.7).

The data from Figure 4.7 (Dagnew et al., 2009) has been reformatted, where relevant, in the following figures for direct comparison with *CFX* RANS results.

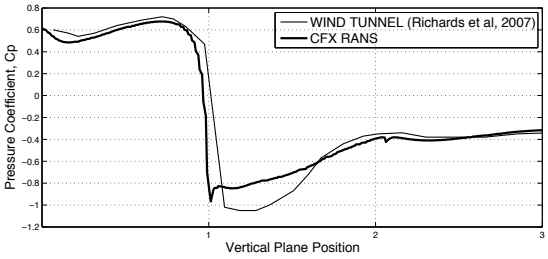
In the following figures, errors between LES and wind-tunnel were measured directly.



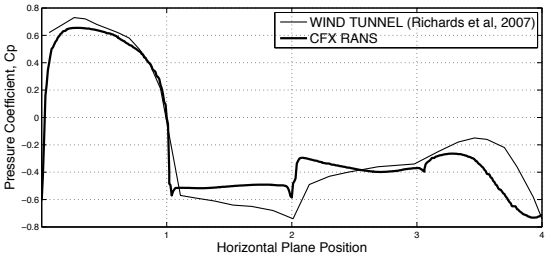
(a) 90°, vertical plane



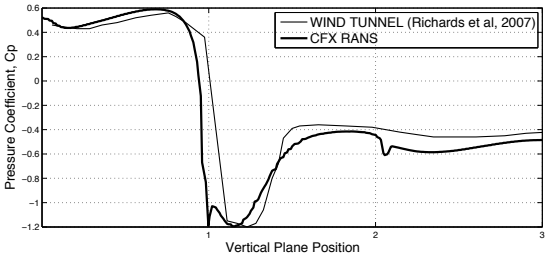
(b) 90°, horizontal plane



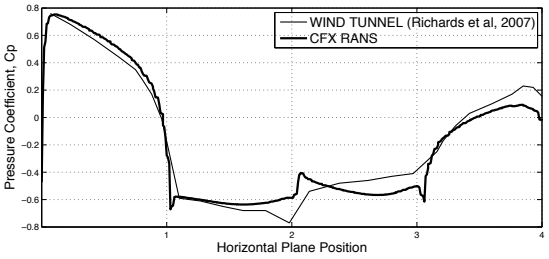
(c) 75°, vertical plane



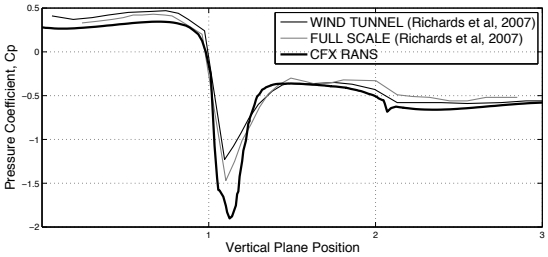
(d) 75°, horizontal plane



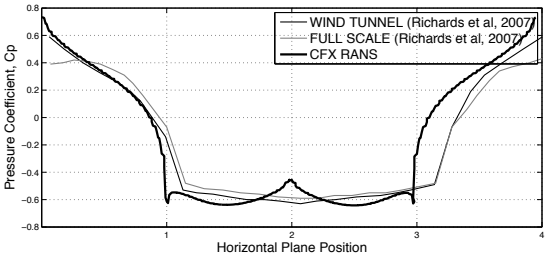
(e) 60°, vertical plane



(f) 60°, horizontal plane



(g) 45°, vertical plane



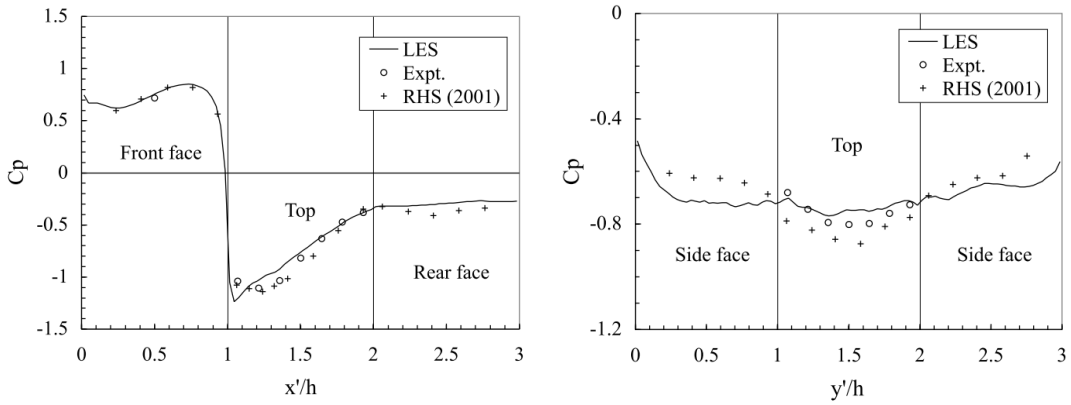
(h) 45°, horizontal plane

Figure 4.8: Cube comparison at 90 - 45° orientation.

Relative errors and accuracies are calculated starting from wind-tunnel vs. full-scale validation as the initial ground-truth. Notice that the direct comparison and relative errors for WT vs. full-scale are the same. Secondly, RANS and LES relative errors are calculated as the direct comparison error plus the ground-truth error, i.e. RANS vs. WT + WT vs. full-scale etc.

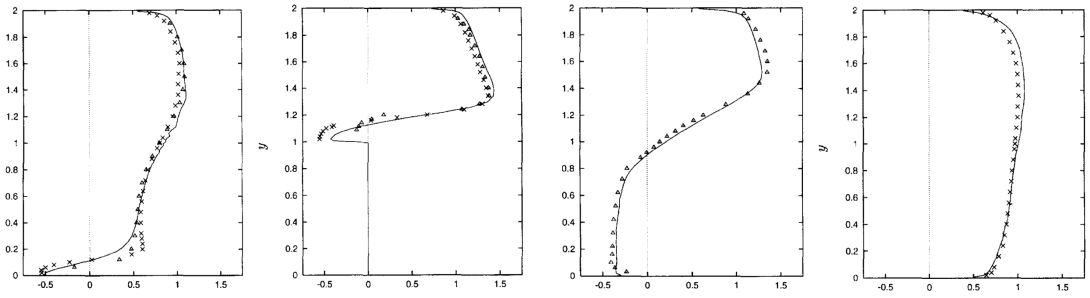
Where comparisons are made from the literature, data sources are specifically noted; where RANS simulation has been undertaken, the source is noted as *CFX*.

Times are derived as follows (in absence of data from original sources): where RANS simulation has been undertaken in *CFX*, the time is simply measured; for similar RANS studies from the



(a) Vertical plane (b) Horizontal plane

Figure 4.9: Surface-mounted cube comparison of LES (-) and wind-tunnel (o) at 90° orientation (Lim et al., 2009).



(a) X=2.68 (b) X=3.50 (c) X=4.56 (d) X=9.00

Figure 4.10: Surface-mounted cube flow field comparison of LES (-) and wind-tunnel (x,Δ) at 90° orientation (Shah & Ferziger, 1997).

literature, the time measured in *CFX* is assigned; for LES studies from the literature, a mean time is assigned from similar LES simulations undertaken in *CFX* as part of work in §6.1; and for wind-tunnel studies, a best guess of 24hours (around 3 working days) based on the author's experience is assigned.

Table 4.5: Surface-mounted cube validation time vs. errors [%].

RANS vs. Full and WT			**Error [%]**		**Accuracy [%]**		
Dir. [°]	**Loc.**	**vs.**	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]	**Source**
90	V. plane	Full	5.654	5.654	94.346	1800	Bitsuamlak et al. (2008)
90	V. plane	Full	8.231	8.231	91.769	1800	Wright & Easom (2003)
90	V. plane	Full	3.505	3.505	96.495	1800	*CFX*
90	H. plane	Full	2.204	2.204	97.796	1800	*CFX*
75	V. plane	WT	3.058	5.582	94.418	1546	*CFX*
75	H. plane	WT	3.885	6.409	93.591	1546	*CFX*
60	V. plane	WT	3.693	6.218	93.782	1715	*CFX*
60	H. plane	WT	2.560	5.085	94.915	1715	*CFX*
45	V. plane	Full	3.844	3.844	96.156	1656	*CFX*
45	H. plane	Full	4.391	4.391	95.609	1656	*CFX*
LES vs. Full and WT			**Error [%]**		**Accuracy [%]**		
Dir. [°]	**Loc.**	**vs.**	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]	**Source**
90	V. plane	Full	2.272	2.272	97.728	11640	Richards et al. (2001)
90	H. plane	Full	2.504	2.504	97.496	11640	Richards et al. (2001)
90	V. plane	WT	2.215	4.739	95.261	11640	Lim et al. (2009)

(...continued on next page)

Surface-mounted cube validation time vs. errors [%] (Table continued.)

90	H. plane	WT	1.039	3.563	96.437	11640	Lim et al. (2009)
90	X=2.68	WT	2.555	5.079	94.921	11640	Shah & Ferziger (1997)
90	X=2.68	WT	2.389	4.914	95.086	11640	Shah & Ferziger (1997)
90	X=3.50	WT	3.582	6.107	93.893	11640	Shah & Ferziger (1997)
90	X=3.50	WT	2.774	5.299	94.701	11640	Shah & Ferziger (1997)
90	X=4.56	WT	1.721	4.246	95.754	11640	Shah & Ferziger (1997)
90	X=9.00	WT	1.491	4.016	95.984	11640	Shah & Ferziger (1997)

WT vs. Full			**Error** [%]		**Accuracy** [%]		
Dir. [°]	**Loc.**	**vs.**	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]	**Source**
90	V. plane	Full	2.018	2.018	97.982	86400	Richards et al. (2007)
90	V. plane	Full	2.320	2.320	97.680	86400	Richards et al. (2007)
45	V. plane	Full	2.648	2.648	97.352	86400	Richards et al. (2007)
45	V. plane	Full	1.769	1.769	98.231	86400	Richards et al. (2007)
90	H. plane	Full	3.375	3.375	96.625	86400	Richards et al. (2007)
90	H. plane	Full	3.456	3.456	96.544	86400	Richards et al. (2007)
45	H. plane	Full	2.572	2.572	97.428	86400	Richards et al. (2007)
45	H. plane	Full	2.038	2.038	97.962	86400	Richards et al. (2007)

From all the RANS, LES, and wind-tunnel validation comparisons in this section, the following mean absolute relative errors can be calculated: wind-tunnel error is 2.524% (σ: 0.623%); LES error is 4.274% (σ: 1.220%); and RANS error is 5.112% (σ: 1.710%).

4.2.5 Results: qualitative validation

Qualitative validation is important in relation to the first hypothesis, that is, in assessing the general success of the local reduced-order model approximations relative to the simulated surface pressure. As well as quantitative analysis of the exact pressure coefficients at vertical and horizontal plane intersections, it is also possible to compare results qualitatively, such as the surface pressure distribution or pattern. In Figure 4.11 the pressure on the top face of a surface-mounted cube from *CFX* RANS simulations at 45, 60, 75, and 90° orientations are compared against wind-tunnel and full-scale data from Richards et al. (2007).

For all orientations there is a greater general similarity in pressure distribution between wind-tunnel (top row) and *CFX* RANS (bottom row) than with the full-scale. Except that in the wind-tunnel results (top row), for 90° and 45° wind directions, one would expect to see generally symmetrical patterning (as is present with the full scale and RANS). This may possibly be an artefact of the difference in time-averaging calculations between the three studies.

Validation studies on tall buildings in the literature are relatively limited in number, compared with studies on simpler geometries such as the surface-mounted cube. Figure 4.12 compares surface pressure contours from wind-tunnel data by Tanaka et al. (2012) and *CFX* RANS. There is generally good agreement between the two sources; although as is typical with wind-tunnel studies, technical measurement difficulties around edges and corners causes a lack of data (Figure 4.12a white regions).

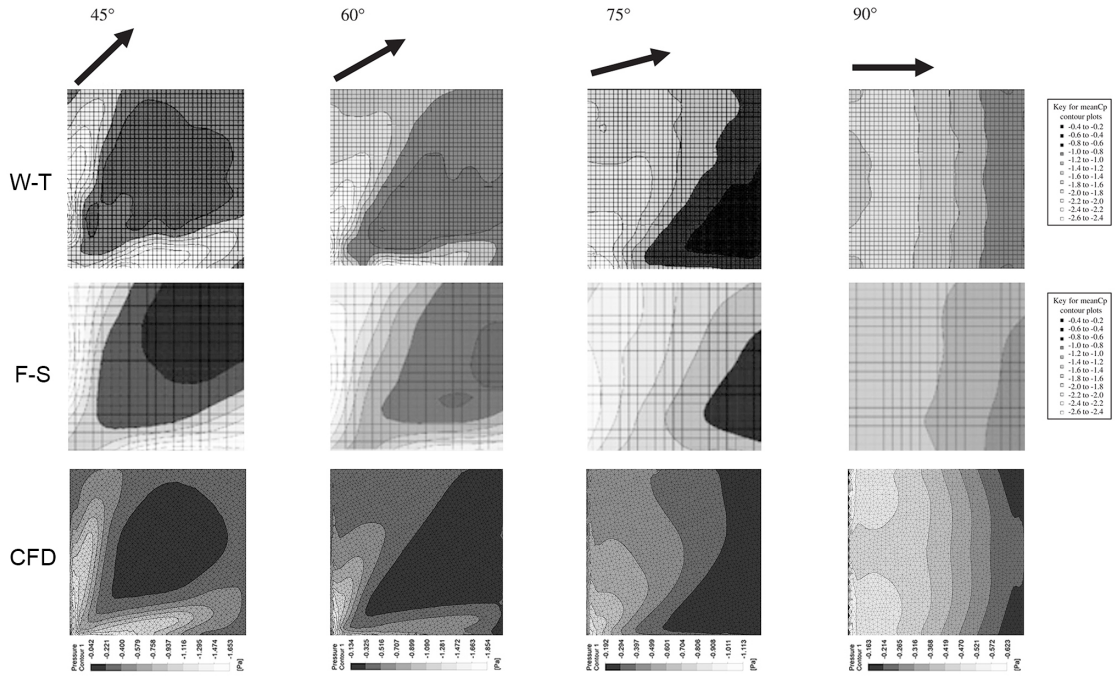
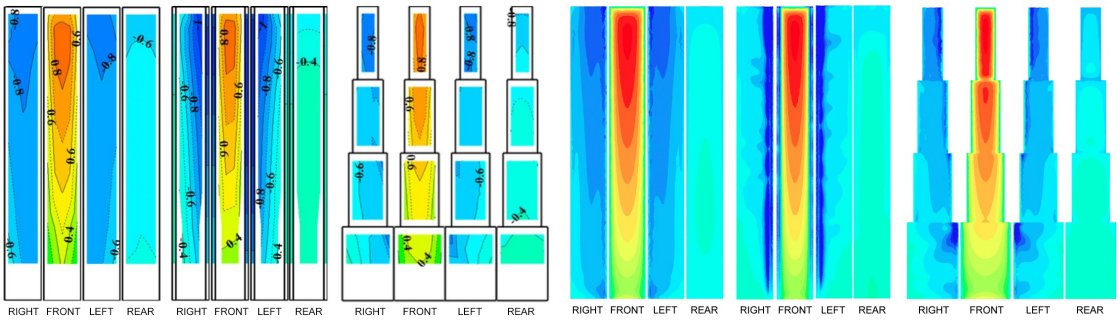


Figure 4.11: Surface-mounted cube mean pressure contours for wind directions 45, 60, 75 and 90°: (top row) wind-tunnel (Richards et al., 2007); (centre row) full-scale (Richards et al., 2007); (bottom row) *CFX* RANS.



(a) (b)

Figure 4.12: Tall building surface pressure: (a) wind-tunnel (Tanaka et al., 2012); (b) *CFX* RANS.

4.3 Fast Fluid Dynamics (FFD) Validation

The fast fluid dynamics (FFD) solver (*'Stable Fluids'* by Stam (1999), §3.2.3) is investigated in this section. Firstly, the solver was developed as an integrated plug-in for *GC* to allow for parametric analysis. Secondly, with this tool a comparative study sought to validate the FFD against CFD to ascertain the difference in results. Finally the validation data from the literature was analysed in detail to also ascertain the difference with CFD and wind-tunnel data.

4.3.1 *GC*-FFD development

The FFD solver was developed to work as a plug-in, or feature node, in *GC*, to take parametric geometry as input and return peak surface pressure. This involved an implementation of the solver with the C# OpenGL library *OpenTK* (Apostolopoulos, n.d.), the code for which can be found in Appendix B.1. The solver itself works as a standalone executable where the *GC* node simply exports the geometry as a list of vertices and indices, instigates the simulation, waits for the result, and parses the result back as a property of that node or geometry (Figure 4.13).

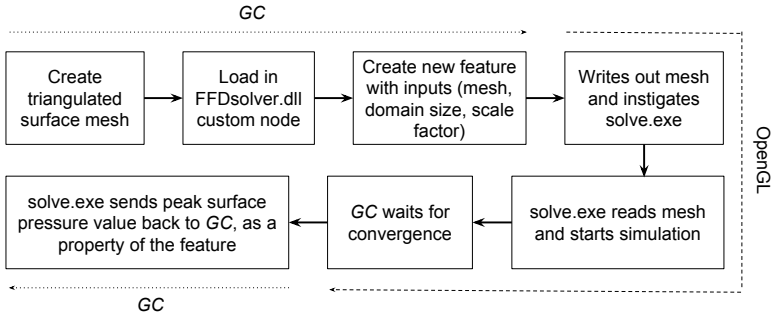


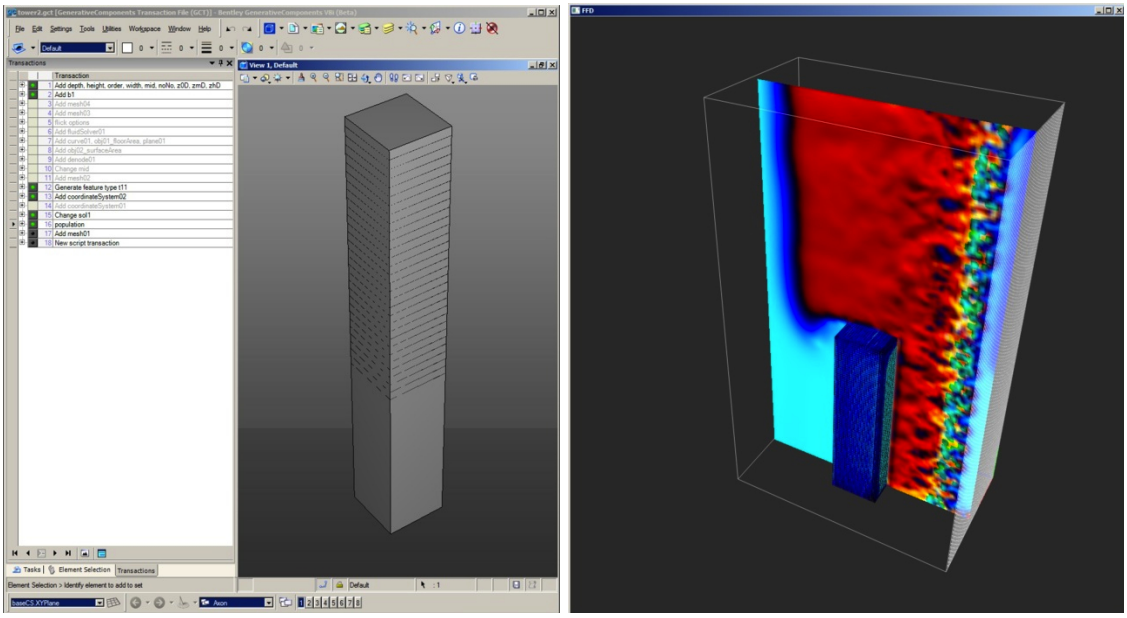
Figure 4.13: *GC*-FFD solver node routine.

The OpenGL executable requires a mesh file and simulation inputs which are created and specified through *GC*. The mesh file read by the FFD solver is a list of voxel centroid points, calculated by using a barycentric coordinate system sampling method over the original surface mesh to get a list of unordered points. These points are then snapped to a structured domain grid resolution, duplicates are removed with a hash set, and then classed as 'solid' for treatment by the solver. The role of *GC* is therefore simply to create the surface geometry, export the mesh, define simulation inputs, and receive the final response from the simulation. The geometry model and FFD simulation are shown in Figure 4.14.

The OpenGL interface tool has further potential as it allows for any iterative process (such as dynamic relaxation, ray tracing, or optimisation/search algorithms) or visualisation (such as rendering or animations) to be run outside of the dependency graph system.

4.3.2 Methodology

The FFD validation has three parts: the first and second are experimental comparisons between FFD and *CFX*; the third is a meta-analysis of an existing study data comparing FFD, RANS, and wind-tunnel by M. Jin et al. (2013). Although there are validation studies of the FFD in the literature which give a general idea about its accuracy and speed, they do not directly compare with *CFX* RANS and are not for tall building, bluff body flow types. The peak pressure was used in some cases because there is no direct correspondence between the FFD and CFD meshes,



(a) *GC* model (b) FFD simulation

Figure 4.14: *GC*-FFD solver node: screenshot of parametric cuboid study.

therefore comparison of exact positions and values is therefore not possible. This is also why the pressure coefficient has not been used for comparison, since there is uncertainty in the other physical variables in the FFD. Results have been normalised because there is no quantifiable relation, and so can only be used to test for similar trends, not exact results. The FFD and CFD results are normalised by the minimum and maximum range of the CFD for each individual case.

4.3.3 Results: single-variate

A parametric model was created in *GC* that allowed control over five basic building parameters: *height*, *width*, *depth*, *orientation*, and *number of edges* (Table 4.6 for parameter ranges). Each parameter was varied individually, starting from a base case design of $width = 10m$, $depth = 10m$, $height = 50m$, $orientation = 0°$, and $number\ of\ edges = 4$, i.e. a cuboid.

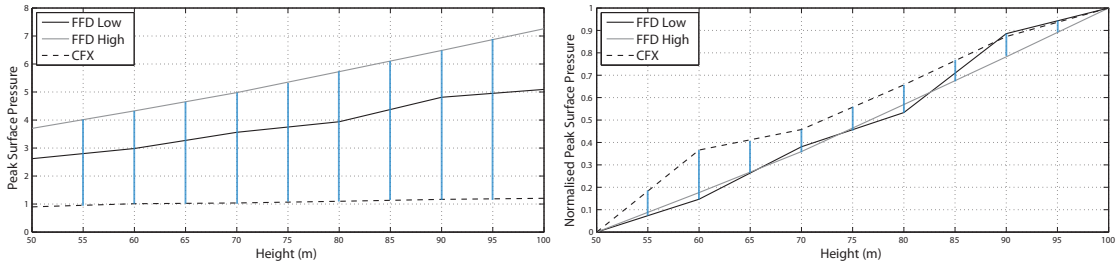
After each model instance is generated, it is exported to the FFD or *CFX* RANS for simulation and the peak surface pressure $[Pa]$ is extracted. The FFD was run for two different resolutions, 'low' and 'high': the low resolution is configured to run in real-time as might be typically used in practice.

Table 4.6: FFD vs. *CFX* single-variate comparison: parameter ranges.

Parameter	Minimum	Maximum	Increment
Height $[m]$	50	100	10
Width $[m]$	10	50	10
Depth $[m]$	10	50	10
Orientation [°]	0	90	15
No. Edges	3	6	1

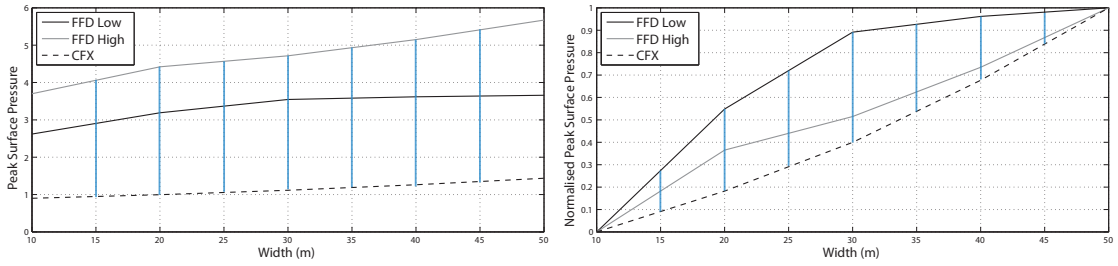
The size of the domain is kept constant at $150m$ across-wind, $150m$ vertically and $200m$ streamwise. A scale factor of 1.0 means the domain has 150 by 150 by 200 cells. The FFD is run at two resolution: the low version with a scale factor of 0.4, i.e. a domain resolution of 60 by 60 by 80 voxels; and high with a scale factor of 0.6, or 90 by 90 by 120 voxels. As the scale factor increases, so too does the resolution of the meshing and the simulation time. At 0.4 (low) the convergence time is less than 30 seconds, at 0.6 (high) is 2 minutes, and at 0.8 it is around 5 minutes. This difference in resolution, the approximation caused by meshing, can be seen in Figure 4.22.

Figures 4.15 to 4.19 shows a breakdown for each parameter against peak surface pressure: (left) raw data; (right) normalised; and for *height*, *width*, *depth*, *orientation*, and *number of edges*.



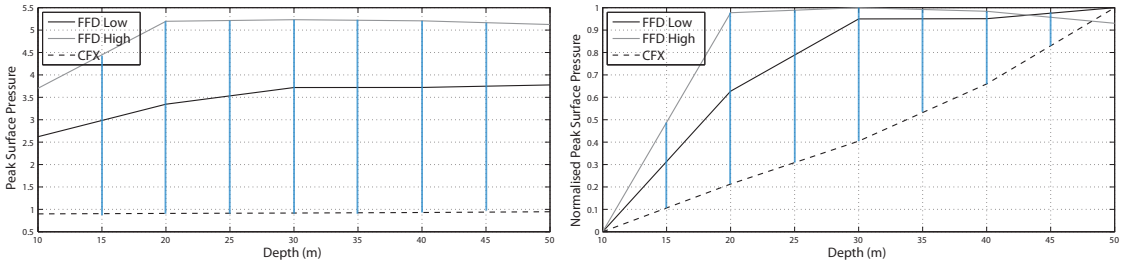
(a) Raw (b) Normalised

Figure 4.15: FFD vs. *CFX* single-variate comparison: *height*.



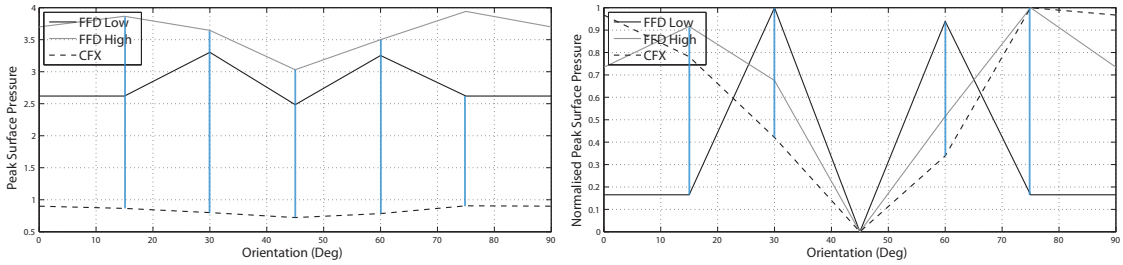
(a) Raw (b) Normalised

Figure 4.16: FFD vs. *CFX* single-variate comparison: *width*.



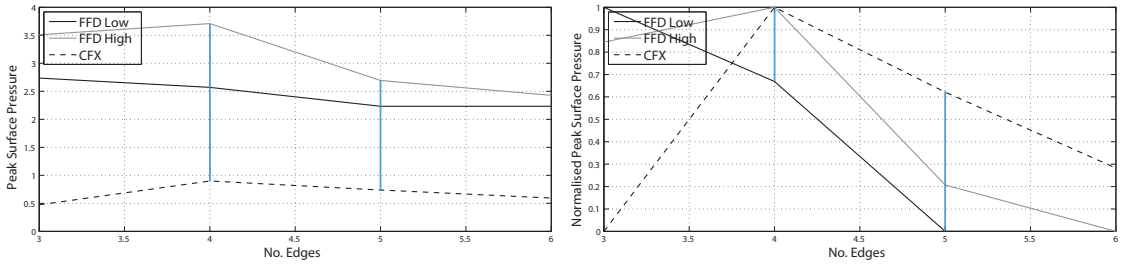
(a) Raw (b) Normalised

Figure 4.17: FFD vs. *CFX* single-variate comparison: *depth*.



(a) Raw (b) Normalised

Figure 4.18: FFD vs. *CFX* single-variate comparison: *orientation*.

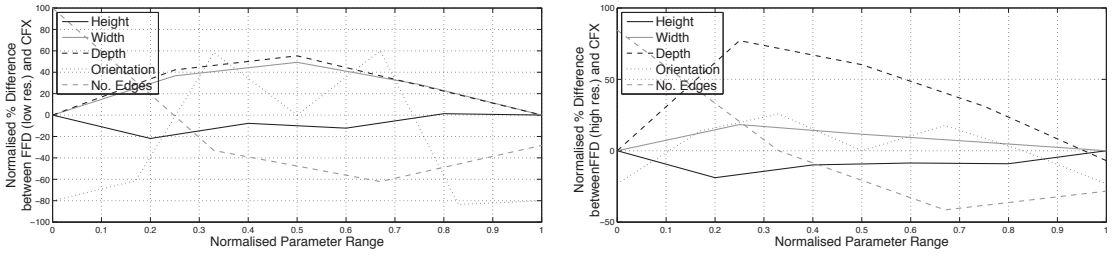


(a) Raw (b) Normalised

Figure 4.19: FFD vs. *CFX* single-variate comparison: *no. edges*.

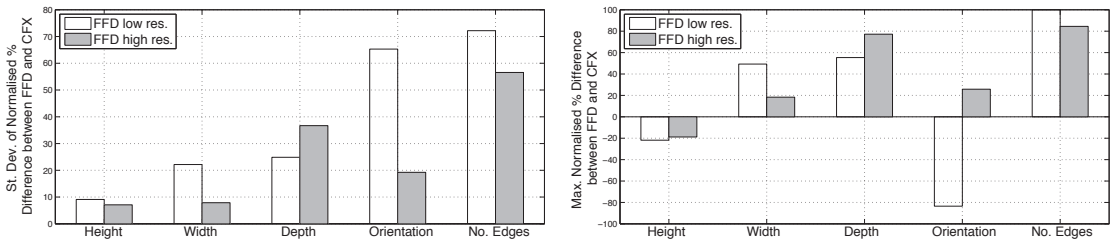
The following observations can be made from the single-variate FFD study: as a trend, the higher resolution FFD has less difference with the *CFX* than the lower resolution one. the higher resolution FFD has a greater difference than the lower resolution one for *Depth*, perhaps because the small variance amplifies the difference; there is a greater difference for *Orientation* and *Number of Edges*, than for *Height* and *Width*, because they both deviate from the orthogonality of the domain grid which introduces inaccuracies in the mesh.

Figures 4.20 and 4.21 summarise all the data for each of the five parameters, comparing the FFD and *CFX* peak surface pressures. Figure 4.20 takes each parameter, across its normalised range, against the normalised % difference between FFD and *CFX*, for (a) low and (b) high FFD resolutions. Figure 4.21 shows the (a) standard deviation and (b) maximum normalised % difference between the FFD (low and high resolutions) and *CFX*.



(a) FFD low resolution (b) FFD high resolution

Figure 4.20: FFD vs. *CFX* single-variate comparison: normalised error [%].

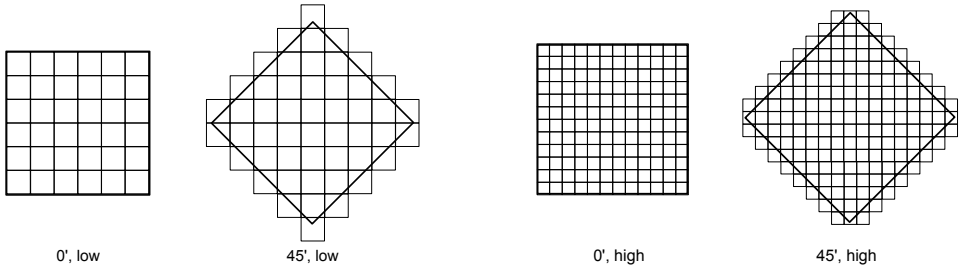


(a) Normalised σ error [%] in peak pressure (b) Normalised max. error [%] in peak pressure

Figure 4.21: FFD vs. *CFX* single-variate comparison: normalised error (σ [%] and max.[%]).

4.3.4 Results: super-ellipse

One of the issues with the FFD is in the meshing approximation: it uses a simple (but fast) voxelisation to represent the surfaces and therefore the accuracy of the representation is heavily dependent on the resolution or size of the voxels (Figures 3.1, 4.14, and 4.22). Since the voxels are cubic they are better at representing orthogonal geometry than sloped or curved surfaces.



(a) (b)

Figure 4.22: FFD voxel mesh resolution: (a) low; (b) high.

The relationship between curvature and representation is analysed through the use of a super-elliptic plan shaped extrusion. Using this topology means it is possible to continuously range from a square through to a circle (Figure 4.23), and then compare results from the FFD (at varying resolutions) to the *CFX* (which uses more advanced surface meshing). The parametric form of a super-ellipse is:

$$x(\theta) = |cos\theta|^{\frac{2}{n}} \cdot a \cdot sgn(cos\theta) \qquad y(\theta) = |sin\theta|^{\frac{2}{n}} \cdot b \cdot sgn(sin\theta) \tag{4.3}$$

where θ ranges from 0 to 90 (quarter profile is simply copy-rotated), a and b are the x and y radii of 10, and n ranges from 1.0 (square) to 2.0 (circle). Any value of n greater than 1 gives a convex profile. *sgn* is the sign, or signum, function (i.e. returns the sign (-1, 0, or +1) of a real number). See Figure 4.23 for the development from square to circular profile.

Figure 4.24 shows the change in peak surface pressure as the section changes from a square (degree 1.0) to a circle (degree 2.0). At the lower resolution the change in section of the model is not

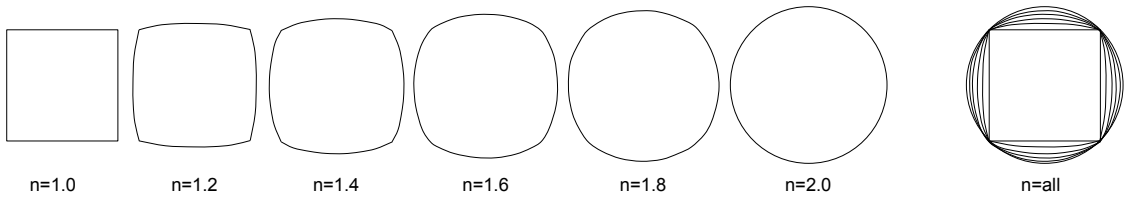
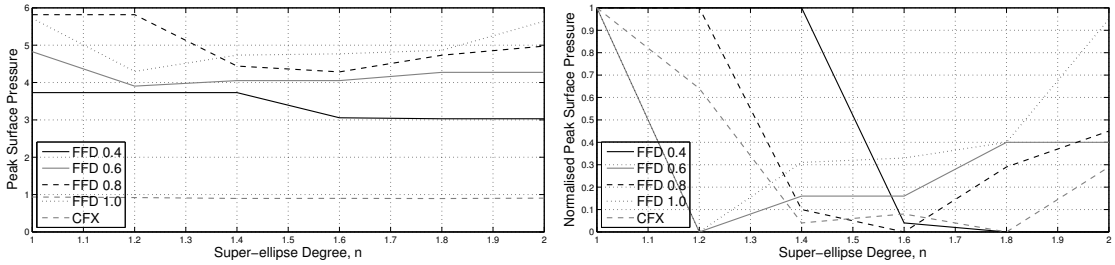


Figure 4.23: Super-ellipse range of horizontal sections.

represented by the mesh, i.e. at a resolution of 40% (0.4) the super-ellipses of degree 1.0 to 1.4 are identical, explaining why there are data points that are exactly the same. On the whole there is poor correlation between any of the FFD resolutions and *CFX*.



(a) (b)

Figure 4.24: FFD vs. *CFX* super-ellipse comparison.

4.3.5 Results: field meta-analysis

The following analysis is based on data from the study by M. Jin et al. (2013) (§3.2.6, Figures 3.13 to 3.15) Cases are given of a wind-driven naturally ventilated room with three configurations (Figure 4.25): i) upstream single-sided opening; ii) downstream single-sided opening; and iii) double-sided opening. For each case, results were obtained from FFD, RANS, and wind-tunnel at four vertical line positions on the stream-wise plane X in the fluid field, moving progressively downstream: $-H/25$; $H/2$; $H+H/25$; $H+H/2$.

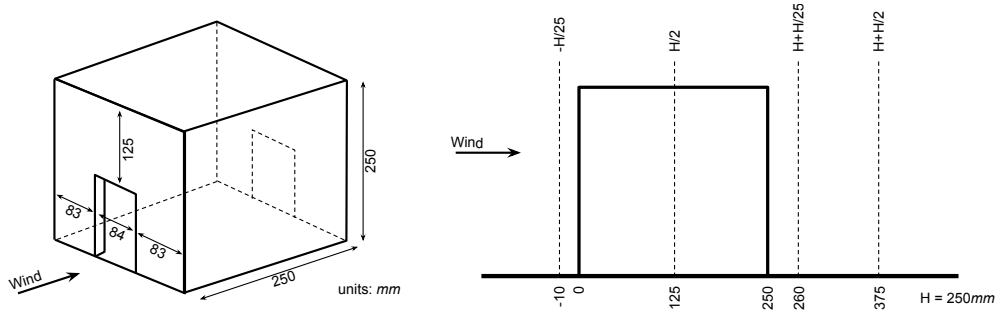
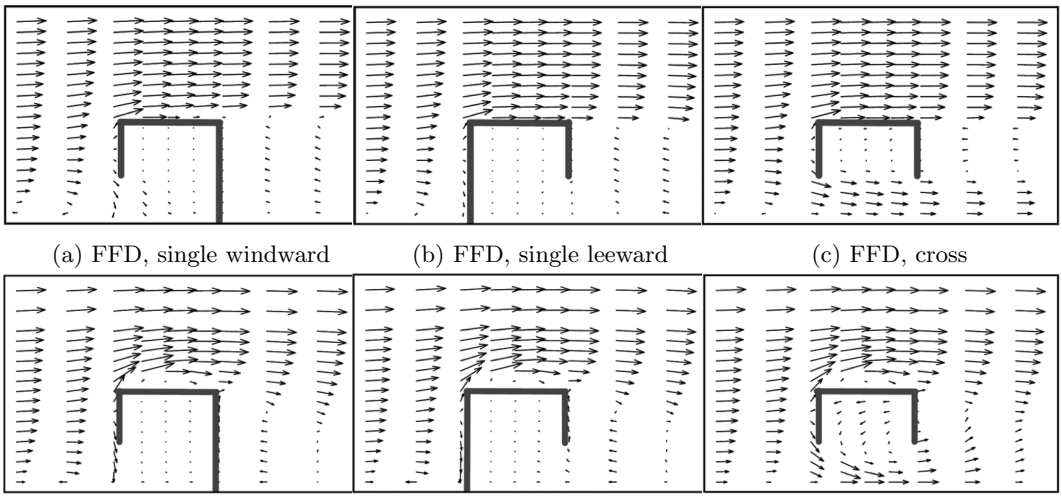


Figure 4.25: FFD validation scale model (left) and evaluation positions (right) (M. Jin et al., 2013).

The field velocity measurements at the stream-wise mid-section plane are shown in Figure 4.26: the FFD on the upper row; the wind-tunnel on the lower row; and the three cases (i, ii, and iii)

in each column. It is clear from these images that the FFD under-predicts around the edges of the models, an indication that there is an issue with the turbulence treatment in the flow and particularly close to surfaces.



(a) FFD, single windward (b) FFD, single leeward (c) FFD, cross

(d) BLWT, single windward (e) BLWT, single leeward (f) BLWT, cross

Figure 4.26: FFD vs. BLWT field comparison at stream-wise mid-section (M. Jin et al., 2013).

Figures 4.27 to 4.29 give the field velocity (U/U_{ref}) comparison measurements between FFD, CFD, and wind-tunnel for the three cases: upstream single-sided opening (Figure 4.27); downstream single-sided opening (Figure 4.28); and double-sided opening (Figure 4.29). The four columns are the four vertical lines on the centre stream-wise plane ($-H/25$, $H/2$, $H + H/25$, $H + H/2$) as shown in Figure 3.11.

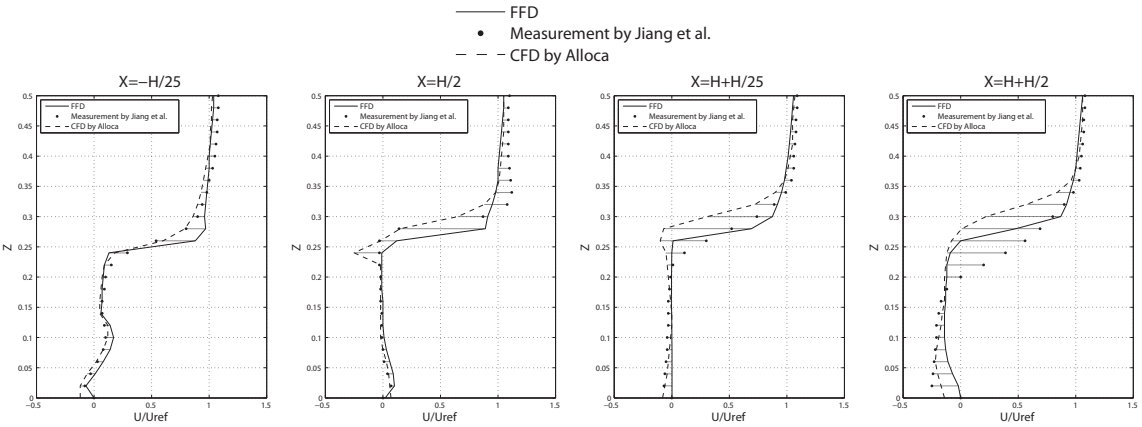


Figure 4.27: FFD vs. WT and CFD field comparison (M. Jin et al., 2013): upstream single-sided.

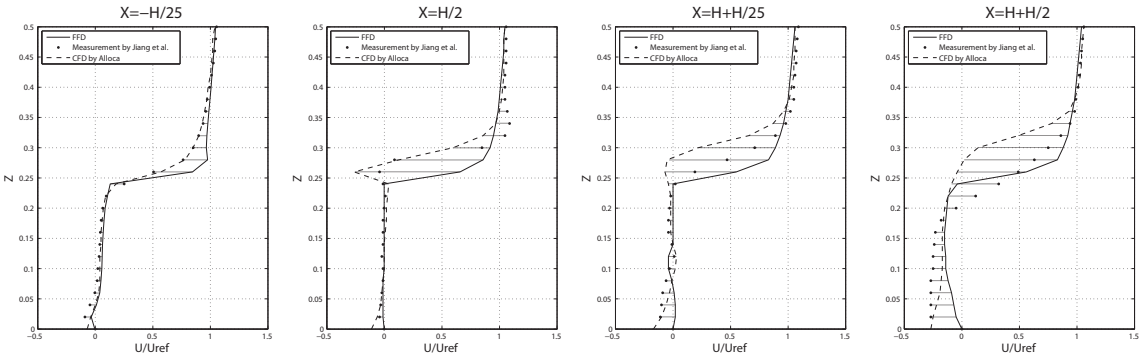


Figure 4.28: FFD vs. WT and CFD field comparison (M. Jin et al., 2013): downstream single.

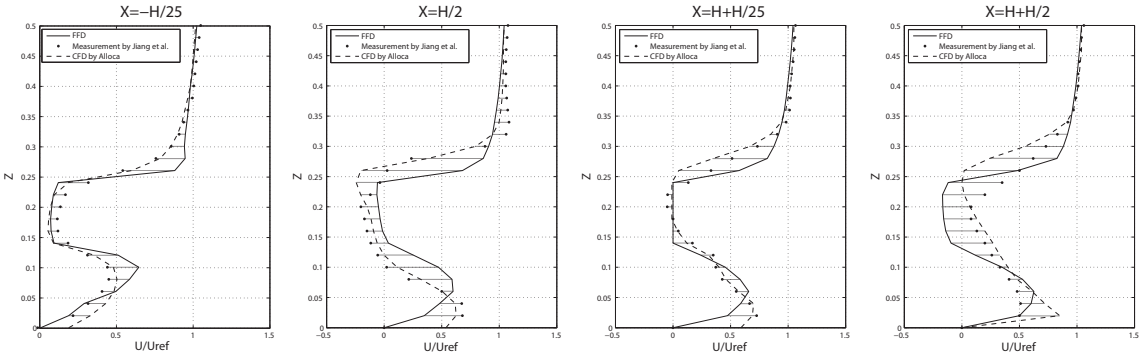


Figure 4.29: FFD vs. WT and CFD field comparison (M. Jin et al., 2013): double-sided.

The minimum, maximum, absolute mean, and standard deviation errors were calculated for each case, for FFD vs. wind-tunnel (Table 4.7) and FFD vs. RANS (Table 4.8). Note that the minimum error is effectively the maximum negative error, i.e. an under-prediction, rather than the maximum error which is an over-prediction.

Table 4.7: FFD validation errors [%]: FFD vs. wind-tunnel.

Upstream single-sided opening (Fig. 4.27)	**X = -H/25**	**X = H/2**	**X = H+H/25**	**X = H+H/2**
Minimum	-14.22	-12.53	-24.65	-41.73
Maximum	30.5	64.65	14.7	16.63
Absolute mean	5.14	7.09	5.26	8.9
Standard deviation	8.17	14.44	7.45	13.64
Downstream (Fig. 4.28) single-sided opening	**X = -H/25**	**X = H/2**	**X = H+H/25**	**X = H+H/2**
Minimum	-11.21	-10.16	-4.64	-27.18
Maximum	31.33	68.93	30.18	16.5
Absolute mean	4.99	7.83	6.42	7.28
Standard deviation	7.69	18.88	9.56	9.86
Double-sided opening (Fig. 4.29)	**X = -H/25**	**X = H/2**	**X = H+H/25**	**X = H+H/2**
Minimum	-20.98	-25.72	-22.73	-47.91
Maximum	36.37	50.88	27.46	21
Absolute mean	8.66	13.58	7.89	12.22
Standard deviation	12.36	18.76	10.98	17.48

Table 4.8: FFD validation errors [%]: FFD vs. RANS.

Upstream single-sided opening (Fig. 4.27)	**X = -H/25**	**X = H/2**	**X = H+H/25**	**X = H+H/2**
Minimum	-4.41	-4.31	-2.09	-1.82
Maximum	24.99	65.57	65.53	49.25
Absolute mean	4.56	7.13	7.3	7.49
Standard deviation	5.87	14.3	15.61	12.27
Downstream single-sided opening (Fig. 4.28)	**X = -H/25**	**X =H/2**	**X = H+H/25**	**X = H+H/2**
Minimum	-4.9	-3.27	-6.21	-1.72
Maximum	25.53	85.56	73.61	61.25
Absolute mean	4.07	8.75	10.72	11.16
Standard deviation	6.11	21.54	20.06	18.13
Double-sided opening (Fig. 4.29)	**X = -H/25**	**X =H/2**	**X = H+H/25**	**X = H+H/2**
Minimum	-19.64	-57.52	-53.08	-36.72
Maximum	30.09	94.65	46.86	60.13
Absolute mean	6.89	17.59	9.88	15.77
Standard deviation	10.52	27.27	17.89	23.28

The errors compared with both wind-tunnel and RANS are typically greater at $X{=}H/2$ and $X{=}H{+}H/2$, i.e. inside the building and in the far wake. These are two regions in which greatest turbulence might be expected; again, lack of robust turbulence treatment with the FFD being the probable cause of the higher errors.

The mean, upper and lower error range across the three cases and the four locations are then calculated for FFD vs. wind-tunnel (Table 4.9) and for FFD vs. RANS (Table 4.10).

Table 4.9: FFD validation errors [%] summary: FFD vs. wind-tunnel.

	Mean	**Upper**	**Lower**
Minimum	-21.97	-4.64	-47.91
Maximum	34.09	68.93	14.7
Absolute mean	7.94	13.58	4.99
Standard deviation	12.44	18.88	7.45

Table 4.10: FFD validation errors [%] summary: FFD vs. RANS.

	Mean	**Upper**	**Lower**
Minimum	-16.31	-1.72	-57.52
Maximum	56.92	94.65	24.99
Absolute mean	9.28	17.59	4.07
Standard deviation	16.07	27.27	5.87

As expected, there is a lower error and variability for FFD vs. wind-tunnel (absolute mean 7.94% σ:12.44%) as compared to FFD vs. RANS (absolute mean 9.28% σ:16.07%). In fact, the errors from this study are less than the other studies. The potential reasons being that: i) the resolution given in the field study by M. Jin et al. (2013) is higher and does not operate in real-time; ii) the FFD solver is better suited to capturing field velocity than to surface pressure due to lack of proper treatment (structured voxel meshes and no boundary layers); and iii) the previous studies calculate peak surface pressure, a single value as compared to the broader field analysis.

4.3.6 FFD validation summary

The results from all of the FFD validation studies are compiled in Table 4.11. Direct comparison mean absolute errors ($\overline{|\delta|}$) [%], mean relative errors against ground-truths (Rel. $\overline{|\delta|}$) [%], mean relative accuracies against ground-truths (100 - Rel. $\overline{|\delta|}$) [%], and simulation times [s] are given for each case.

The direct comparison errors are simply comparing RANS CFD or wind-tunnel, and are therefore only partial errors. The ground-truths used to calculate the relative error and relative accuracy when comparing the FFD with wind-tunnel and RANS are calculated in the previous section. For wind-tunnel the ground-truth error is 2.525% and for RANS it is 5.112%. The relative error is simply calculated by adding the direct error to the appropriate ground-truth; the relative accuracy is also then simply just the inverse of the error, i.e. relative accuracy equals 100%-relative error.

Table 4.11: FFD validation time vs. errors [%].

FFD vs. WT,	**Error [%]**		**Accuracy [%]**	
Meta field study	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]
Up. s-s opening, X = -$H/25$	5.140	7.664	92.336	131.56
Up. s-s opening, $X = H/2$	7.090	9.614	90.386	131.56
Up. s-s opening, $X = H$+$H/25$	5.260	7.784	92.216	131.56
Up. s-s opening, $X = H$+$H/2$	8.900	11.424	88.576	131.56
Down s-s opening, $X = H$+$H/2$	4.990	7.514	92.486	131.56
Down s-s opening, $X = H/2$	7.830	10.354	89.646	131.56
Down s-s opening, $X = H$+$H/25$	6.420	8.944	91.056	131.56
Down s-s opening, $X = H$+$H/2$	7.280	9.804	90.196	131.56
D-s opening, X = -$H/25$	8.660	11.184	88.816	131.56
D-s opening, $X = H/2$	13.580	16.104	83.896	131.56
D-s opening, $X = H$+$H/25$	7.890	10.414	89.586	131.56
D-s opening, $X = H$+$H/2$	12.220	14.744	85.256	131.56

FFD vs. RANS	**Error [%]**		**Accuracy [%]**	
Meta field study	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]
Up. s-s opening, X = -$H/25$	4.560	9.672	90.328	131.56
Up. s-s opening, $X = H/2$	7.130	12.242	87.758	131.56
Up. s-s opening, $X = H$+$H/25$	7.300	12.412	87.588	131.56
Up. s-s opening, $X = H$+$H/2$	7.490	12.602	87.398	131.56
Down s-s opening, $X = H$+$H/2$	4.070	9.182	90.818	131.56
Down s-s opening, $X = H/2$	8.750	13.862	86.138	131.56
Down s-s opening, $X = H$+$H/25$	10.720	15.832	84.168	131.56
Down s-s opening, $X = H$+$H/2$	11.160	16.272	83.728	131.56
D-s opening, X = -$H/25$	6.890	12.002	87.998	131.56
D-s opening, $X = H/2$	17.590	22.702	77.298	131.56
D-s opening, $X = H$+$H/25$	9.880	14.992	85.008	131.56
D-s opening, $X = H$+$H/2$	15.770	20.882	79.118	131.56

FFD vs. RANS,	**Error [%]**		**Accuracy [%]**	
single-var. (high resolution)	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]
h=60m	18.903	24.015	75.985	120
h=70m	9.919	15.031	84.969	120
h=80m	8.619	13.731	86.269	120
h=90m	9.137	14.249	85.751	120
w=20m	18.317	23.429	76.571	120
w=30m	11.575	16.687	83.313	120
w=40m	5.904	11.016	88.984	120
d=12m	33.508	38.620	61.380	120
d=16m	86.937	92.049	7.951	120

(...continued on next page)

FFD validation time vs. errors [%] (Table continued.)

d=20m	77.137	82.249	17.751	120
d=30m	60.322	65.434	34.566	120
d=40m	31.161	36.273	63.727	120
d=50m	6.989	12.101	87.899	120
o=0°	23.138	28.250	71.750	120
o=15°	13.840	18.952	81.048	120
o=30°	25.761	30.873	69.127	120
o=60°	17.666	22.778	77.222	120
o=90°	23.138	28.250	71.750	120
n=3	84.453	89.565	10.435	120
n=5	41.465	46.577	53.423	120
n=6	28.345	33.457	66.543	120

FFD vs. RANS,	**Error [%]**		**Accuracy [%]**	
single-var. (low resolution)	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]
h=60m	21.848	26.960	73.040	30
h=70m	7.780	12.892	87.108	30
h=80m	12.231	17.343	82.657	30
h=90m	1.228	6.340	93.660	30
w=20m	36.623	41.735	58.265	30
w=30m	49.218	54.330	45.670	30
w=40m	28.572	33.684	66.316	30
d=20m	42.139	47.251	52.749	30
d=30m	55.304	60.416	39.584	30
d=40m	27.863	32.975	67.025	30
o=0°	80.070	85.182	14.818	30
o=15°	61.313	66.425	33.575	30
o=30°	58.253	63.365	36.635	30
o=60°	60.072	65.184	34.816	30
o=75°	83.496	88.608	11.392	30
o=90°	80.070	85.182	14.818	30
n=3	100	100	0	30
n=4	33.135	38.247	61.753	30
n=5	62.168	67.280	32.720	30
n=6	28.345	33.457	66.543	30

FFD vs. RANS,	**Error [%]**		**Accuracy [%]**	
super-ellipse	$\overline{\lvert\delta\rvert}$	Rel. $\overline{\lvert\delta\rvert}$	100 - Rel. $\overline{\lvert\delta\rvert}$	**Time** [s]
n=1.0, res.=0.4	0.000	5.112	94.888	30
n=1.2, res.=0.4	35.971	41.083	58.917	30
n=1.4, res.=0.4	95.815	100	0	30
n=1.6, res.=0.4	4.620	9.732	90.268	30
n=1.8, res.=0.4	0.000	5.112	94.888	30
n=2.0, res.=0.4	28.596	33.708	66.292	30
n=1.0, res.=0.6	0.000	5.112	94.888	120
n=1.2, res.=0.6	64.029	69.141	30.859	120
n=1.4, res.=0.6	11.869	16.981	83.019	120
n=1.6, res.=0.6	7.576	12.688	87.312	120
n=1.8, res.=0.6	39.918	45.030	54.970	120
n=2.0, res.=0.6	11.321	16.433	83.567	120
n=1.0, res.=0.8	0.000	5.112	94.888	300
n=1.2, res.=0.8	35.971	41.083	58.917	300
n=1.4, res.=0.8	6.021	11.133	88.867	300
n=1.6, res.=0.8	8.478	13.590	86.410	300
n=1.8, res.=0.8	29.182	34.294	65.706	300
n=2.0, res.=0.8	16.497	21.609	78.391	300
n=1.0, res.=1.0	0.000	5.112	94.888	900
n=1.2, res.=1.0	64.029	69.141	30.859	900
n=1.4, res.=1.0	26.556	31.668	68.332	900
n=1.6, res.=1.0	24.654	29.766	70.234	900
n=1.8, res.=1.0	39.952	45.064	54.936	900
n=2.0, res.=1.0	65.919	71.031	28.969	900

From all of the studies on FFD, the mean absolute error is 21.016% (σ:31.544%). It can be observed

that increased resolution greatly affects the simulation speed, meaning to get a more reliable result (or as good as the solver will allow by reducing the error in the meshing) it is prohibitively slow. Developing a better mesh (i.e. unstructured) would probably improve the accuracy but also impact the simulation speed. The absence of a turbulence model is also a source of concern: without a turbulence model its use is limited to predicting generalised flow patterns rather than fine scale details. Zuo & Chen (2009) tested the addition of a turbulence model but found it less accurate than without one, suggesting the development of a specific turbulence model for the FFD is also required. Use of a structured (voxelised) mesh as well as a lack of robust wall treatment creates challenges in comparing surface pressures with traditional CFD and wind-tunnel measurements. In conclusion, all of these factors make the FFD more suited to velocity field analysis of flows with low turbulence.

4.4 Summary

The purpose of this chapter was to quantify the speeds and relative accuracies of the FFD, RANS, LES, and wind-tunnel approaches against a ground-truth. These existing tools are used to measure the success of the proposed methodology. The mean absolute accuracies and times of the FFD, RANS, LES and wind-tunnel against full-scale ground truth data are shown in Figure 4.30.

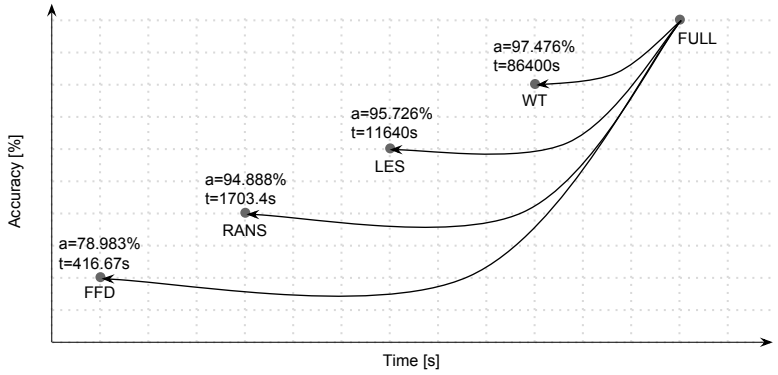


Figure 4.30: Accuracy [%] and time [s] of FFD, RANS, LES, and wind-tunnel.

The mean absolute ground-truth errors are as follows: Fast Fluid Dynamics 21.0% (σ=31.5%); CFD RANS 5.1% (σ=1.7%); CFD LES 4.3% (σ=1.2%); wind-tunnel 2.5% (σ=0.6%). These errors are purely based on the data within this chapter and are therefore only reflective of these studies. It was not a primary objective of this thesis to conduct extensive CFD validation studies; however the ground-truth errors here are indicative of the relative order and basic position of the various approaches in the time-accuracy domain.

Chapter 5

Approximations

An overview of the individual studies included in this chapter is given in Table 5.1. These progress from a non-CFD trivial example cause, to methodological sensitivity analyses, through to a series of cases of increasing complexity testing the fundamental principles of the approach. The chapter concludes with a brief summary, however general observations are collected in Chapter 7.

Table 5.1: Summary of Chapter 5 studies.

	Study	**Alg.**	**Output**	**Section**
Global shape				
	Real buildings	LS-SVM	Global peak	5.2
Local shape				
	Introductory case	ANN	Vertex	5.3.2
	Feature sensitivity	RF	Vertex	5.3.5
	Hidden layer sensitivity	ANN	Vertex	5.3.6
	ANN generalisability	ANN/RF	Vertex	5.3.7
	Cuboid orientation	ANN	Vertex	5.3.8
	Cuboid height	ANN	Vertex	5.3.9
	Topology	ANN	Vertex	5.3.10
	Concave super-formula	ANN	Vertex	5.3.11
Local shape with interference				
	Single upstream cuboid interference	ANN	Vertex	5.4.2
	Multiple cuboid interference	ANN	Vertex	5.4.3

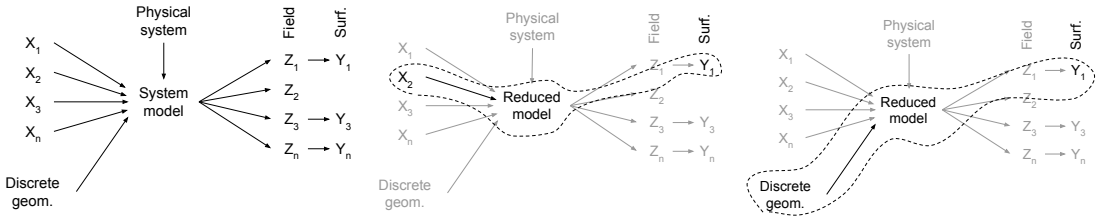
5.1 Reduced-Order Modelling

In the previous chapter, the speed and accuracy of the FFD and CFD were assessed, making it apparent that a new approach with the speed of FFD and the accuracy of CFD would be beneficial. This breaks away from the current trend of having either a fast *or* accurate approach, to pursue a novel approach which attempts to achieve both. To improve simulation time whilst retaining accuracy, machine learning is proposed to approximate CFD through model reduction. The process can be generally described as:

$$f : \mathbf{X} \rightarrow Y \tag{5.1}$$

where: $\mathbf{X}$ is the input vector of prediction features; Y is the output response; and f comprises the regression function or reduced-order model (Figure 5.1). The reduced-order model focuses on a small portion of the full system by selecting input-output relationships of interest. Typically, $\mathbf{X}$

consists of boundary condition inputs (Figure 5.1b, such as Mach number or angle-of-attack for airfoils). In the proposed definition, the relationship is between discrete (meshed) model geometry as input and surface pressure (Figure 5.1c). As such, a proportion of the simulation data is discarded when generating the approximation.



(a) Simulation (b) Typical reduction (c) Proposed reduction

Figure 5.1: Reduced-order model schematic.

The reduced-order model f is generated with training set $\mathbf{S}$ and tested with set $\mathbf{T}$:

$$\mathbf{S}_{(\mathbf{R}^{\mathbf{X}+Y},n)} \qquad \mathbf{T}_{(\mathbf{R}^{\mathbf{X}+(Y)},m)} \tag{5.2}$$

where the dimensionality $\mathbf{R}$ of $\mathbf{S}$ is equal to the input vector, $\mathbf{X}$, plus the output, Y, by the number of samples, n. The dimensionality $\mathbf{R}$ of $\mathbf{T}$ is the same as for $\mathbf{S}$; except the output vector (Y) is kept separate for calculating the prediction error. See §7.3 for further discussion on set dimensions.

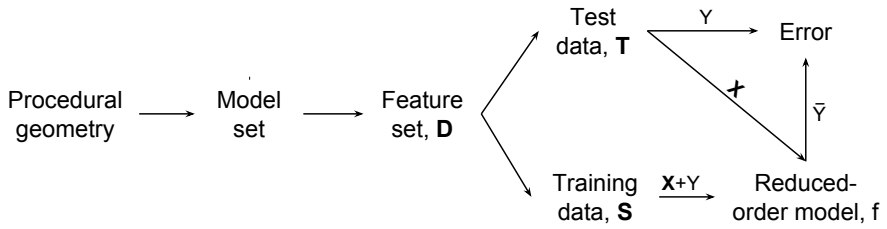
The distinction is drawn between the simulation output response Y from CFD, and the prediction output response Y' from the reduced-order model. For a single sample i, the difference between the two is used to calculate the sample prediction error, δ_i:

$$\delta_i = (\bar{Y}_i - Y_i)/Y_{range} \cdot 100 \tag{5.3}$$

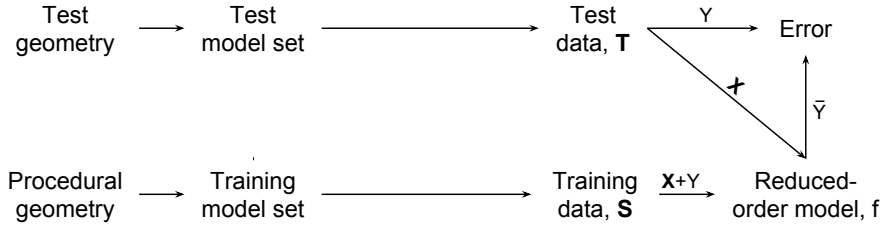
The quantitative statistics used for reporting the error are the:

- real-valued minimum $\delta_{min.}$ and maximum $\delta_{max.}$ of the error range, i.e. the worst-case vertex predictions;
- mean of the absolute error range, $\overline{|\delta|}$. Note, the mean absolute error range is not equal to the absolute mean error range, i.e. $\overline{|\delta|} \neq |\bar{\delta}|$ or $mean(abs(\delta)) \neq abs(mean(\delta))$;
- standard deviation of the absolute error range, $\sigma_{|\delta|}$. Note, although the standard deviation is always positive, the standard deviation of the absolute error range is also not equal to the standard deviation of the error range, i.e. $\sigma_{|\delta|} \neq \sigma_{\delta}$.

There are two types of test used in the following chapters: sample-based (Figure 5.2a); and model-based (Figure 5.2b).



(a) Sample-based



(b) Model-based

Figure 5.2: Sample- and model-based testing.

In the sample-based test, m and n are drawn from the same set of available data D, meaning that $m = D - n$. m and n are both randomised in this case, and are used to monitor error convergence and assess the learning process. For model-based tests, a completely different test set is used so that m and n are independent, such as in the case where a procedural model is used for training and real models for testing; model-based tests are used for assessment of the hypotheses.

The sample-based test is essentially the ROM training error since they are usually presented at the converged sample size for each case. Whereas the model-based test is a more practical demonstration of the realistic predictive accuracy on a new, unseen case. As such, the sample-based errors are typically lower than the model-based ones.

A key aspect of model reduction is in the definition of the input and output feature vector, i.e. identifying the characteristics of the simulation that are of interest whilst discarding the rest. In this chapter, three definitions are tested: peak pressure prediction from global parameters; local pressure prediction from shape features; and local pressure prediction from shape and fluid features. Each is a development and improvement from the last in some respect, allowing for either a more flexible reduced-order model, greater output, or for greater problem complexity. Similarly, it is only meaningful to calculate prediction times for model-based tests.

5.2 Global Prediction

The first definition is the simplest in that it follows a conceptually conventional, top-down approach, where an entire building model is characterised by global parameters such as its height, width, or orientation. These properties give a rough characterisation of the overall form from which other

general properties can be predicted. In this case, the global parameters are matched with the peak surface pressure.

This section describes the use of a procedural tall building model and a least-square support vector machine (LS-SVM) (Suykens et al., 2011) to predict the peak surface pressure. The model is validated against real examples to measure its success at representing complex geometries and its predictive accuracy.

5.2.1 Methodology

The methodology is divided into four parts: procedural geometry generation (§5.2.2); CFD evaluation of the training set (§5.2.3); test set evaluation (§5.2.4); and learning with the LS-SVM (§5.2.5). The whole process is summarised in Figure 5.3, split into the training (left) and test (right, dashed) stages.

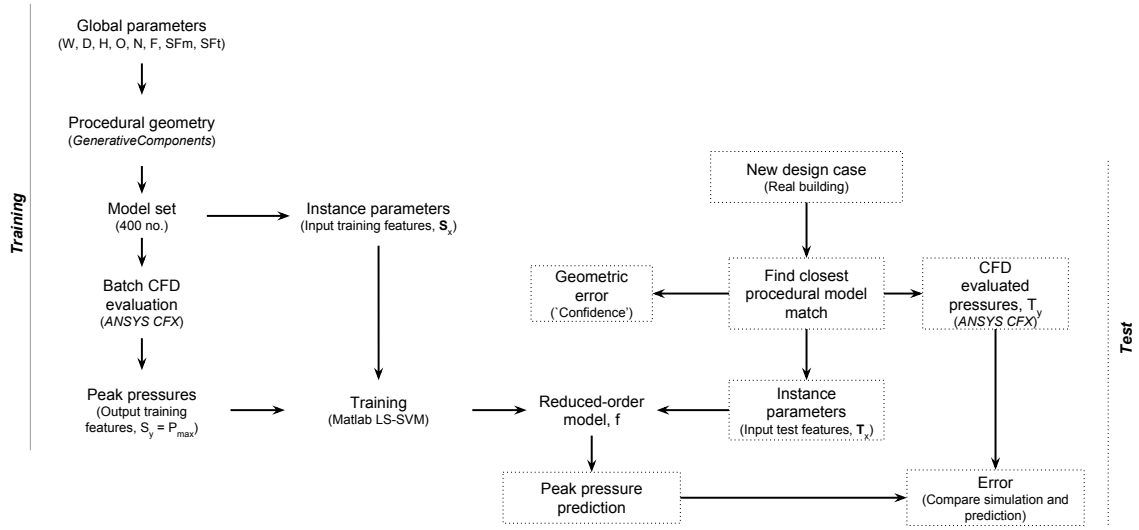


Figure 5.3: Global prediction procedure summary.

5.2.2 Methodology: procedural tall building model

A parametric model was created in *GC* with the goal of creating a generalised tall building model, with two objectives which basically define the efficiency of the model. The two contradictory objectives are to minimise the number of parameters whilst maximising the design representation potential, i.e. the number of possible buildings it could create. The aim is therefore to simply have the maximum output from the minimum input.

Minimising the number of parameters becomes a significant problem for sampling; the 'curse of dimensionality' describes the combinatorial relationship $n^{\mathbf{R}}$, where n is the number of increments per parameter and $\mathbf{R}$ is the number of parameters, or the dimensionality. For example, two

parameters each with three increments gives $3^2 = 9$ samples, and three parameters each with three increments gives 27 samples. At the increment resolutions and ranges given in Table 5.2, the procedural model has around $5.99e^{15}$ (5990 trillion) different potential combinations. In the end only 400 samples (model instances) were used for training, equal to $6.68e^{-12}$% of the potential samples. The second objective of the procedural model is to maximise its flexibility in matching with potential realistic designs, assessed in §5.2.4.

5.2.3 Methodology: training set simulation

The procedural model is used to generate 400 model instances by randomly sampling the defined parameter space, storing the instance's parameters for each to use as input vector **X**.

Table 5.2: Procedural tall building model parameters and ranges.

Parameter	Minimum	Maximum	Increment	Potential instances
Width, W $[m]$	5.0	50.0	0.1	451
Depth, D $[m]$	5.0	50.0	0.1	451
Height, H $[m]$	50.0	100.0	0.1	501
Orientation, O [°]	0.0	180.0	0.1	1801
No. edges, N	3	6	1	4
Fillet, F $[m]$	0.0	5.0	0.1	51
Vertical mid-point scale factor, SF_M	0.5	1.5	0.1	16
Vertical top-point scale factor, SF_T	0.1	1.0	0.1	10
			Total	$5.99e^{15}$

Each model must then be evaluated[1] as described in §4.2.1 (the entire evaluated set is shown in Figure 5.4). Each simulation requires around 30 minutes, totalling around 200 hours to evaluate the entire training set of 400 models.

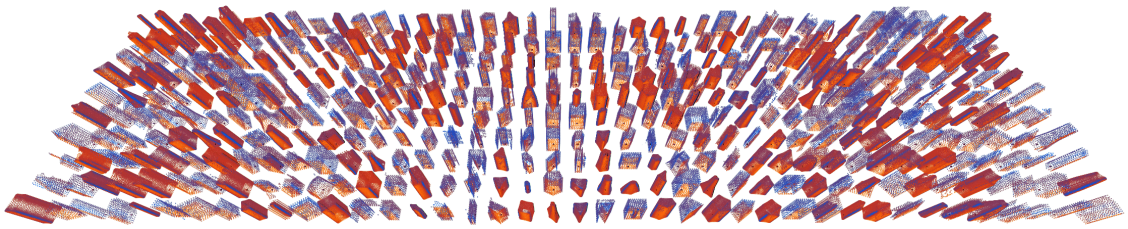


Figure 5.4: Training set of 400 procedural models evaluated with *CFX*.

For each model, the peak (spatially, not in time) surface pressure is found for use as the training output Y, so that the training set is now:

$$\mathbf{S}_{(8+1,400)} \tag{5.4}$$

5.2.4 Methodology: real building test set

The test set of seven real buildings are shown in Figure 5.5 and detailed in Table 5.3. Selection of buildings for the test set is determined by the uniqueness of the design relative to others, therefore

[1]Batch process is initiated through the *CFX* console with *FORFILES /M *.def /C "cmd /c cfx5solve -def @file"*

each of the buildings has at least one architectural (geometric) feature that is unique.

Table 5.3: Real building test set details.

Case	Name	Location	Height $[m]$	Completion date
1	Met Life Building	New York City, US	246.3	1963
2	The Shard	London, UK	306.0	2013
3	Willis Tower (Sears)	Chicago, US	442.1	1974
4	Euston Tower	London, UK	124.0	1970
5	Taipei 101	Taipei, Taiwan	508.0	2004
6	Shanghai World Financial Centre	Shanghai, China	492.0	2008
7	Bank of China Tower	Hong Kong, China	367.4	1990

In Figure 5.5 the top row shows the model as extracted from *Google Earth* and rebuilt in *GC*; and the lower row shows the model vertices (black points), the vertices from the closest matching procedural model (red points), and the difference between closest vertices (red to green lines). Since all of the test models are above $100m$ in height, and the procedural model height was limited to between 50 and $100m$, the test models were all scaled to have a maximum height of $100m$. Whilst this is not consistent with best simulation practice, within these tests the difference is irrelevant.

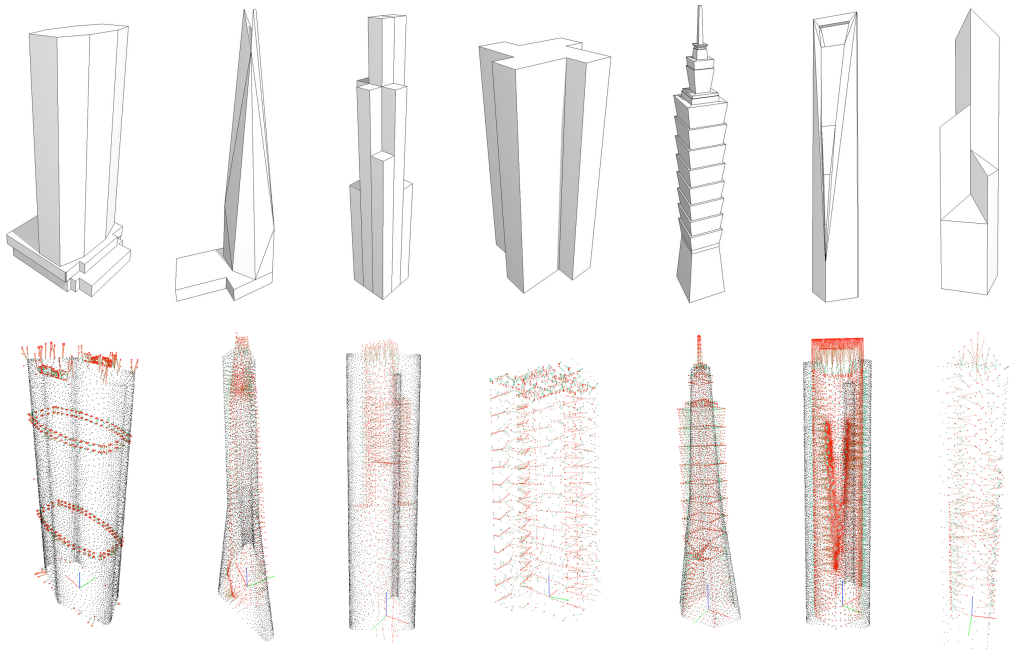


Figure 5.5: Test set of real buildings: (upper) *GC* models; (lower) best procedural model matches.

This can be assessed by comparing a selection of real building models with the best match from the procedural model set. The real building test models were extracted from *Google Earth* and imported into *GC* to be reconstructed. For each, the closest matching procedural model is found by calculating the average distance between the two vertices of the two meshes:

$$\bar{d} = \frac{1}{n} \cdot \Sigma_{i=0}^{n}(|\mathbf{a}_i - \mathbf{b}|) \tag{5.5}$$

where $\bar{d}$ is the mean match difference, i is the vertex index of the test model, n is the number of vertices on the test model, $\mathbf{a}_i$ is a test model vertex, and $\mathbf{b}$ is the closest point on a procedural model. The procedural model was randomly varied until $\bar{d}$ converges and the closest match found.

The results of this matching give the efficiency and effectiveness of the procedural model, as well as identifying design aspects that are currently not included. For example, the model does not currently include vertical rotational twisting and will not give a good match for test models with this feature. Although not implemented here, the representation error can also give the level of confidence to assign to a prediction. Figure 5.6 shows the mean mesh differences for the test set. In the selection, the mean distance ranges between 1.35 and 3.34m.

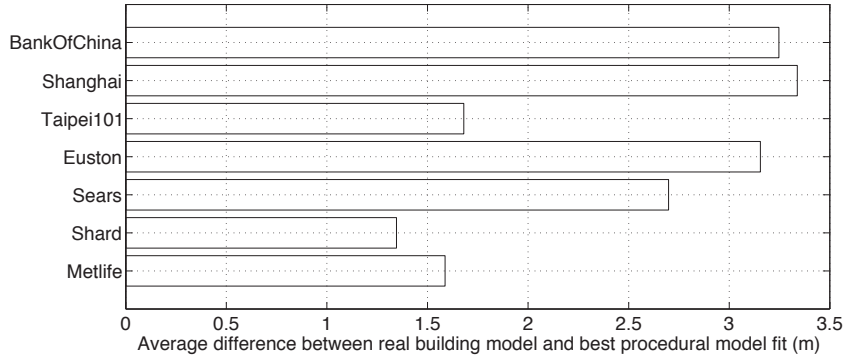


Figure 5.6: Mean difference [m] between real building model and closest procedural model.

5.2.5 Methodology: support vector machine (SVM)

In this section, a support vector machine algorithm is used for the model reduction, as introduced in §3.3.4. The peak surface pressure P_{max} is defined by the parameters given in Table 5.2, giving the following model definition:

$$f^{SVM} : (W, D, H, O, N, F, SF_M, SF_T) \rightarrow P_{max} \qquad \mathbf{S}_{(8+1,400)} \qquad \mathbf{T}_{(8+(1),7)} \qquad (5.6)$$

For the model-based results, the training set $\mathbf{S}$ consists of 400 samples with 8 input features $\mathbf{X}$ and 1 output Y; and the test set $\mathbf{T}$ has 7 samples with the same number of features.

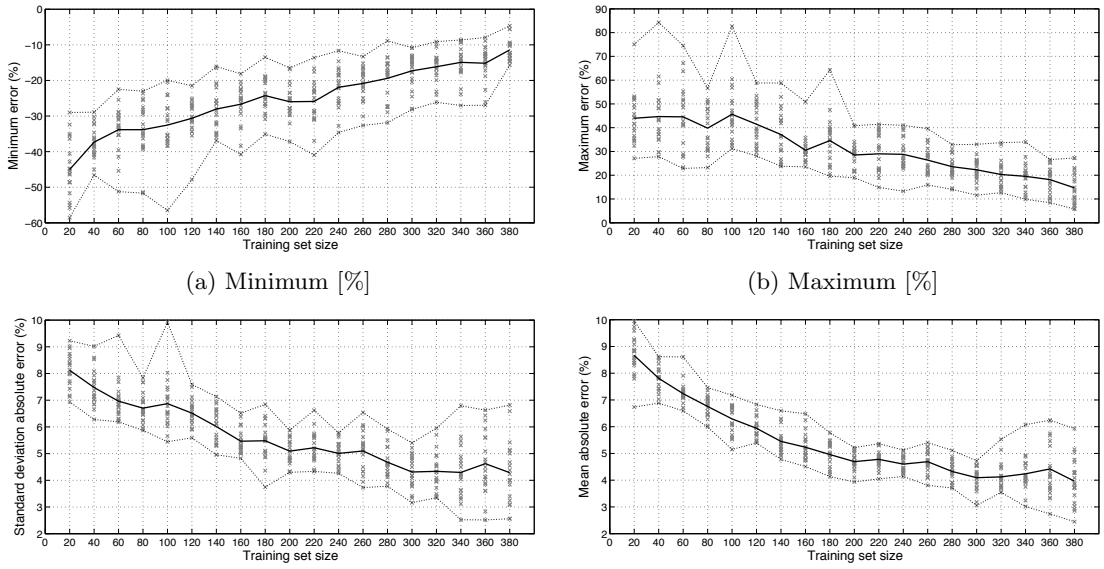
Although suitable for this study, with a relatively small training set size, it becomes evident that the SVM is too slow for the larger sets of subsequent studies. This leads to the potential replacements of the random forest and artificial neural networks; although it is shown that, although both are substantially faster than the SVM, the RF tends to over-train more than the ANN (§5.3.7).

5.2.6 Results

Results: sample-based

In Figure 5.7, out of 400 available samples, the training:test (n:m) set sizes are varied from 10:390 through to 380:20. This set-up highlights: i) the need for adequate training:test set size ratios (variability in standard deviation absolute and mean absolute errors increase significantly after 300:100); ii) and that the global feature definition, due to its costly one-simulation-to-one-sample configuration, means there is a limited set size in comparison to subsequent sections.

The training is run 20-times, i.e. r=20, to allow for variability in the randomised selection of training:test data. The minimum, maximum, standard deviation of the absolute error, and the mean of the absolute error are given in Figure 5.7. In each graph, for each training set size, the grey markers show the error from the 20 runs, dotted lines show the minimum and maximum, and the solid line the mean, of the 20 runs.



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 5.7: Global: sample-based training error convergence.

The variability in the standard deviation and mean absolute errors increases after a training set size of n=300. This is due to the total set size D=400, leaving only a small test set m. The sample-based errors are therefore: min. = -11.417%, max. = 14.666%, mean abs. = 3.960%, and σ abs. = 4.284%.

Results: model-based

For the final test on the set of seven real buildings all 400 samples are used, i.e. n=400 and m=7. Figure 5.8 gives the simulated, Y, and predicted, Y', peak surface pressures [Pa] for the closest

procedural model matches of the seven test case buildings. With the exception of the Metlife and Shard models, Y' is under-predicted.

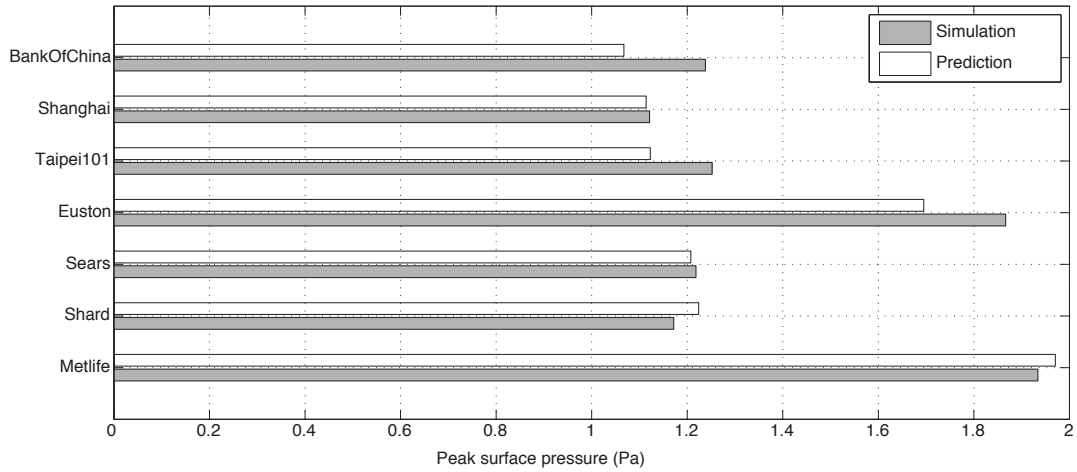


Figure 5.8: Simulated vs. predicted peak surface pressure $[Pa]$ for real building test set.

The prediction errors compared to the simulations range from 6.435% to -21.074%, with a mean absolute error of 10.156% (Figure 5.9).

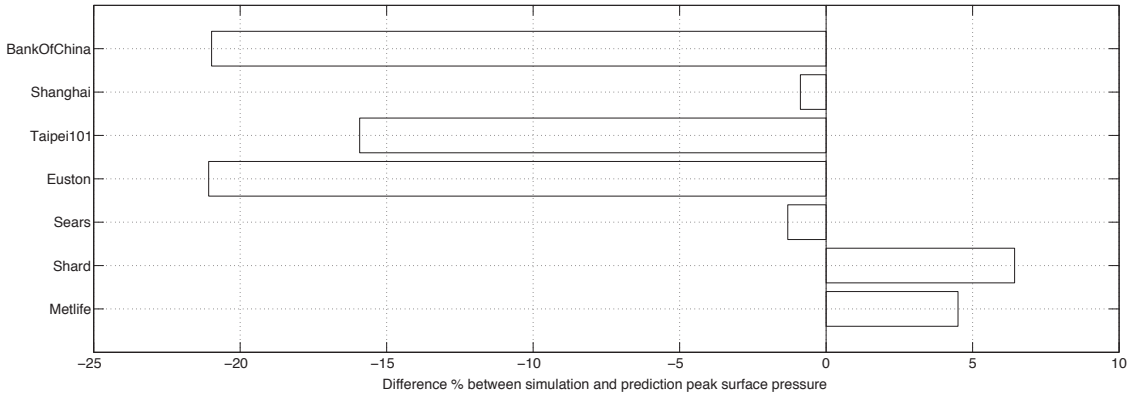


Figure 5.9: Difference [%] of simulated and predicted peak surface pressure for real test set.

Table 5.4: Global: model-based ROM time vs. errors [%].

		Test time $[s]$		
Case	**Error [%]**	**Feature generation**	**Prediction**	**Total**
1	4.504	80.054	0.00449	80.058
2	6.435	75.485	0.00565	75.491
3	-1.308	72.269	0.00481	72.274
4	-21.074	71.863	0.00605	71.869
5	-15.920	72.221	0.00534	72.226
6	-0.876	72.634	0.00404	72.638
7	-20.974	74.410	0.00460	74.415
Min.	-21.074	71.863	0.00404	71.869
Max.	6.435	80.054	0.00605	80.058
Mean Abs.	10.156	74.134	0.00500	74.139
σ Abs.	8.941	2.930	0.00071	2.929

There is a low correlation ($R^2 = 0.265$) between the match difference $[m]$ and the prediction error [%]. This indicates that the two error measurements are relatively independent of one another. Prediction errors and times are given in Table 5.4. In this case the feature generation time is

the match search time, i.e. identifying the closest procedural model and parameters to the test model.

Limitations of this global approach are primarily that it is inflexible, has limited output, and is domain specific (discussion in §5.5). This is a general issue with machine learning in similar situations, identified in the review chapter (§3.3). In the next section, these issues are addressed by redefining the features to be intrinsic and vertex-based, rather than extrinsic and global.

5.3 Shape Features

The methodology in the previous section limited the scope of the problem and learning output, i.e. it was limited by the procedural model and to predicting the peak surface pressure. An improved model definition is proposed here that takes the intrinsic characteristics of a mesh vertex as inputs, such as its position, curvature, and normal, and the vertex pressure as output. It will be demonstrated here through a series of test cases how this definition is far more flexible in both input and output.

5.3.1 Methodology

A general overview of the methodology used in this section is given here (Figure 5.10), with noted differences from the global prediction (Figure 5.3).

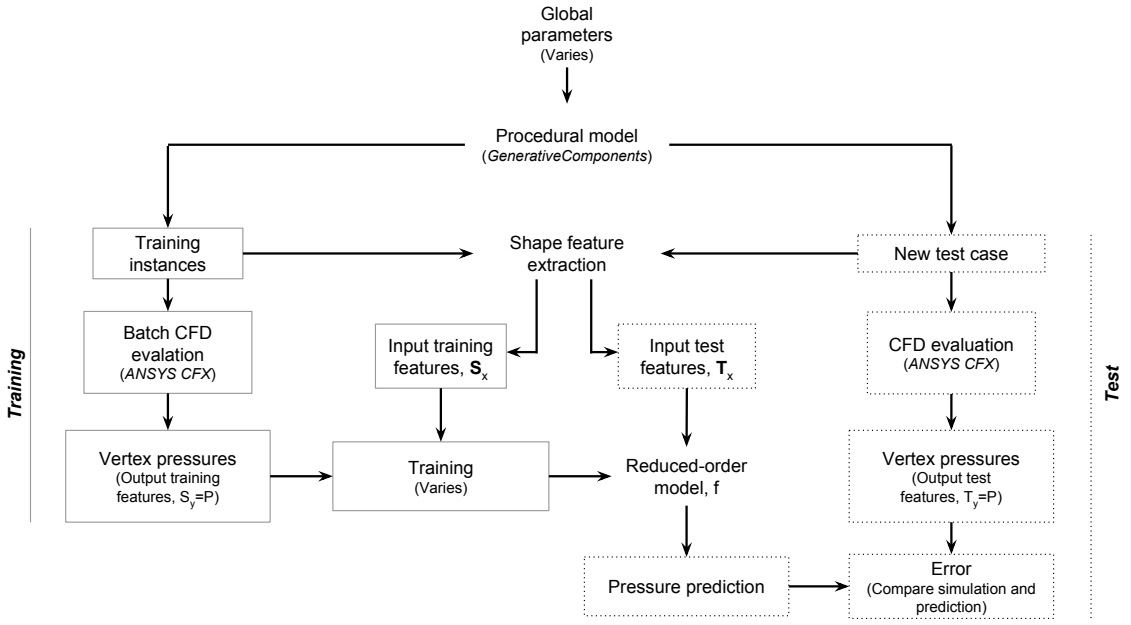


Figure 5.10: Local prediction method: (left, grey boxes) training; (right, dashed boxes) test.

The term 'shape feature' introduced earlier (§3.3.7) refers to the process of deriving each mesh vertex's unique description based on its local geometry. The reduced-order model can now be

redefined in the following way:

$$\text{(General) } f : \mathbf{X} \rightarrow Y \tag{5.7}$$

$$\text{(Global) } f : \text{Global parameters (e.g. } W, D, H) \rightarrow \text{Global metric (e.g. } P_{max}) \tag{5.8}$$

$$\text{(Local) } f : \text{Vertex shape descriptor (e.g. } \mathbf{n}_{x,y,z}) \rightarrow \text{Vertex metric (e.g. } P) \tag{5.9}$$

The change from global model parameters to vertex descriptors results in a large change in input-output resolution and a subsequent increase in training/test set sizes. The increase is in the order of magnitude of three, i.e. each model which previously represented one sample now has thousands of samples ($\mathbf{S}$=400 $\rightarrow$ $\mathbf{S}$=751331, an increase by a factor of 1878 or the mean number of vertices on a single model).

In a change from the previous section, an artificial neural network (ANN) (§3.3.4) is used here. This is due to the increased size of the training and test sets for which the ANN is more amenable. For example, with a training set of 1000 samples the LS-SVM required 183.6s compared to the ANN requiring only 13.7s to generate the reduced-order model (a factor difference of 13.4).

5.3.2 Methodology: example case

A working example test case is used here to demonstrate the principles of the local shape feature approach on a trivial non-CFD problem. A hemisphere was randomly sampled with points (Figure 5.11) and the surface normals calculated for each point. The normal, $\mathbf{n}$, takes the form of a Cartesian triplet of x, y, and z components, so each point has a unique description, $\mathbf{n}_{x,y,z}$, for the input feature vector $\mathbf{X}$.

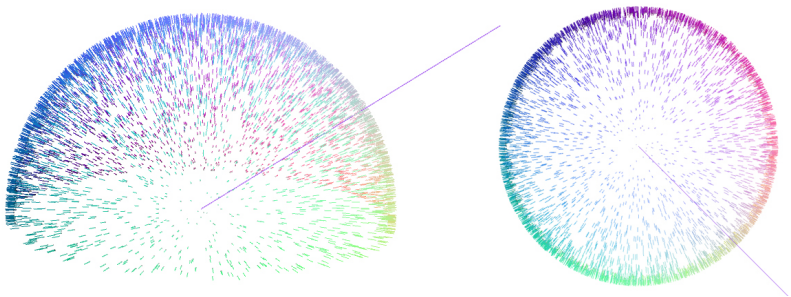


Figure 5.11: Hemisphere $\mathbf{n}_{x,y,z}$ mapped to colour space $\mathbf{C}_{R,G,B}$ with 5000 sample points.

In this case the output, Y, is equivalent to the light intensity (insolation) on the unobstructed hemisphere from a single light source. The reduced-order model definition is therefore:

$$f^{ANN} : \mathbf{n}_{x,y,z} \rightarrow \mathbf{a} \bullet \mathbf{b} \qquad \mathbf{S}_{(3+1,n)} \qquad \mathbf{T}_{(3,5e^4)} \tag{5.10}$$

where $\mathbf{n}_{x,y,z}$ are the normal components of a sample point, $\mathbf{a}$, on the hemisphere, and $\mathbf{a} \bullet \mathbf{b}$ is the

dot product of that point, **a**, and the solar point vector, **b**. The dot product is the insolation, or light intensity, on the surface at point **a** if the light source is at point **b**. The function f then seeks to approximate the light intensity at sample points on the surface given only the normal vector of those points. The case is trivial because **X** is linearly dependent on **Y**.

The number of samples initially generated, n is 5000 (Figure 5.11); however since the problem is relatively simple the number of training samples needed was in fact much less. Figure 5.12 shows the progressive testing, given more training samples, and the convergence on the correct output (far right). After only 50 samples, i.e. $\mathbf{S}_{(3+1,50)}$, the function can be approximated successfully.

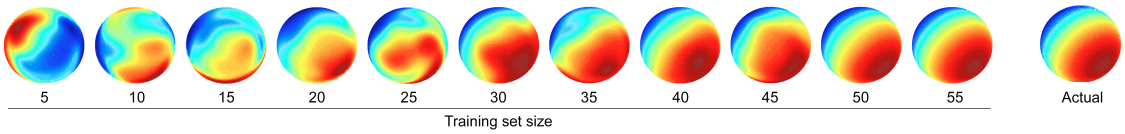
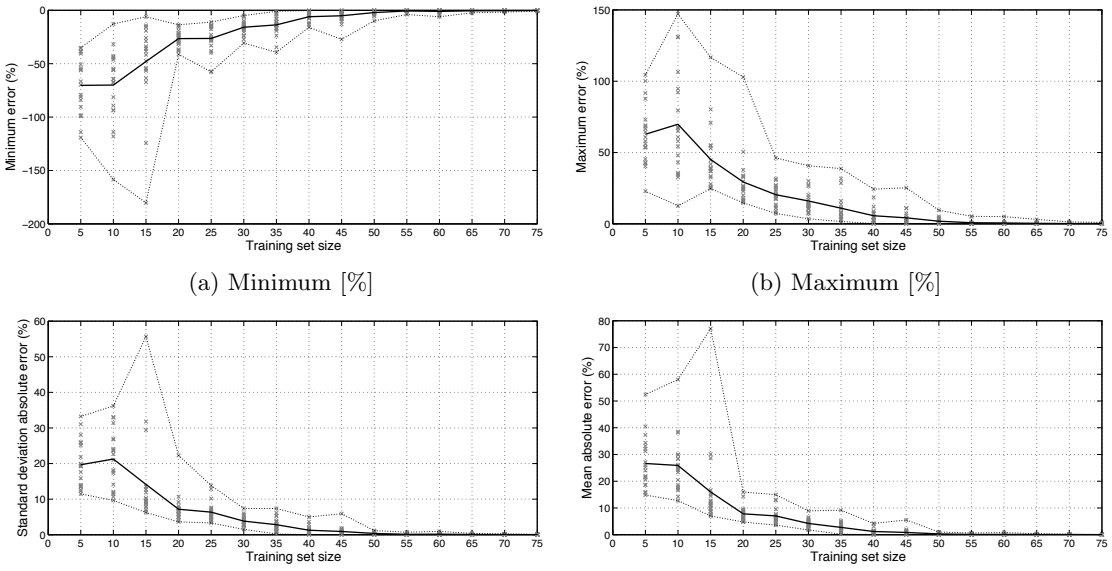


Figure 5.12: Insolation: training convergence, n=5, 10, ..., 55 samples.

Figure 5.13 shows the error convergence as the training set size, n, is increased from 5 to 75. Each shows the training convergence from 20 individual runs (grey points), the error range (dotted lines), and the mean (solid line).



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 5.13: Insolation: sample-based training error convergence.

The errors given are at the converged training set size (Figure 5.13) of n=75. The sample-based errors are therefore: min. = -0.2415%, max. = 0.2271%, mean abs. = 0.0235%, and σ abs. = 0.0334%.

This basic example proves that simple shape features can be used with the ANN to predict another simple geometric metric that is directly dependent on it. In the next development, the number and complexity of input shape features is increased, as well as the adoption of CFD in generating

the output response. The CFD evaluation process is by far the most time-consuming element of the entire process, and the physical relationship that is being learnt or approximated is indirect and non-linear.

5.3.3 Methodology: shape feature definition

The basic concept is to define the pressure at a point on a model by its geometric characteristics, i.e. its location on the model, proximity to an edge, curvature, relative position on the vertical wind profile distribution, and direction of orientation. These features are calculated for every vertex, along with its pressure, to be used as a sample. The definition of the model is now:

$$f^{ANN} : (Z, \mathbf{n}_{x,y,z}, \mathbf{n}\sigma^{1-5}_{x,y,z}, \mathbf{T}_{x,y,z}) \rightarrow P \tag{5.11}$$

- *Height:* Z is simply the height of $\mathbf{V}$, i.e. $\mathbf{V}_z$;
- *Normal:* $\mathbf{n}_{x,y,z}$ are the normal components of $\mathbf{V}$;
- *Curvature:* $\mathbf{n}\sigma^{1-5}_{x,y,z}$ is the standard deviation of the vertex normals in each independent ring, weighted by the inverse of the distance. This is also visualised in Figure 5.14.

$$\mathbf{n}\sigma^{r} = \left(\frac{1}{n-1} \sum_{i=1}^{n} \frac{(\mathbf{n}_i - \bar{\mathbf{n}})^2}{d} \right)^{\frac{1}{2}} \tag{5.12}$$

  Where: r is the vertex neighbourhood ring; n is the number of vertices in r; d is the distance between each vertex in n and the central feature vertex; $\bar{\mathbf{n}}$ is the average of the normals in r; $\mathbf{n}_i$ are all the normals in each neighbourhood ring, r.

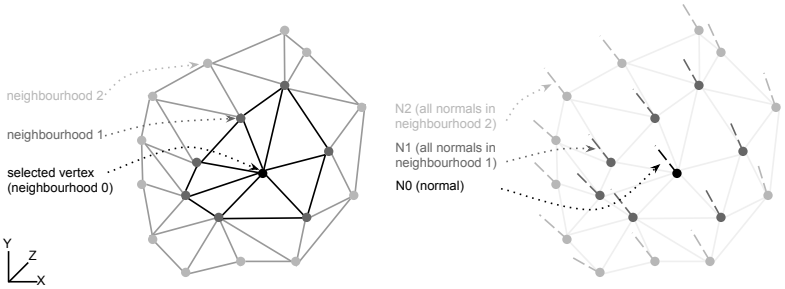


Figure 5.14: Mesh vertex and normal neighbourhoods for curvature analysis.

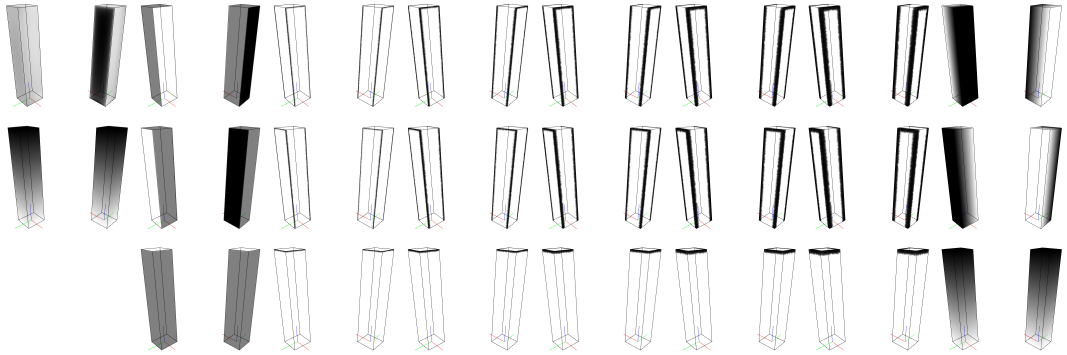
- *Position:* $\mathbf{T}_{x,y,z}$ is the normalised position of $\mathbf{V}_i$ within the range of all model vertices $\mathbf{V}$

$$\mathbf{T}_i = \frac{(\mathbf{V}_i - \mathbf{V}_{min})}{(\mathbf{V}_{max} - \mathbf{V}_{min})} \tag{5.13}$$

- *Pressure:* P is simply the pressure at $\mathbf{V}$ as extracted from the simulation. This can quite easily be replaced with any dependent secondary metric, such as force or the pressure coeffi-

cient.

All of the features are visualised independently in Figure 5.15, in pairs showing the front and back faces of a W:D:H=1:1:5 ratio cuboid. With the exception of the first column, the top row is the x component, central is y, and lowest is z.



(a) P, Z (b) $\mathbf{n}$ (c) $\mathbf{n}\sigma^1$ (d) $\mathbf{n}\sigma^2$ (e) $\mathbf{n}\sigma^3$ (f) $\mathbf{n}\sigma^4$ (g) $\mathbf{n}\sigma^5$ (h) $\mathbf{T}$

Figure 5.15: Local shape feature visualisation: pairs of (left) front and (right) back face.

5.3.4 Methodology: shape feature calculation

The following pseudocode is a simplification of the full code given in Appendix B.3 (the standard deviation calculation code is given in Appendix B.2).

```
for each mesh {
    read each v{x,y,z} {
        calculate minRange{x,y,z}
        calculate maxRange{x,y,z}
        }
    read each i{a,b,c}
    read each n{x,y,z}
    read each P
    for each vertex {
        for each neighbourhood ring (0 to 5) {
            find indices of connected vertices, e.g. for vertex index a: r0{a}, r1{b,c,d,e,f}, etc.
            }
        for each neighbourhood ring (1 to 5) {
            calculate standard deviation of vertex normals in r1-5
            }
        print vertex feature X{z,n,nσ^1-5,T} and Y{P}
        }
    }
```

The output of the calculation, per vertex, is simply a 23-dimensional vector. E.g. z{0.52}, $\mathbf{n}${-0.68,0.72,-0.05}, $\mathbf{n}\sigma^1${-0.03,-0.03,-0.02}, $\mathbf{n}\sigma^2${-0.07,-0.07,-0.03}, $\mathbf{n}\sigma^3${-0.11,-0.10,-0.02}, $\mathbf{n}\sigma^4${-0.15,-0.14,-0.02}, $\mathbf{n}\sigma^5${-0.20,-0.18,-0.03}, $\mathbf{T}${0.79,0.13,0.04}, P{-0.17}.

The feature generation time, for both training and test models, is currently 0.02784 s/sample. For example, a model with a mesh of 1000 vertices currently requires 27.84s. The feature calculation times for different neighbourhood sizes is given in Table 5.6 and Figure 5.16, with the mean, σ,

minimum and maximum for 10 re-runs. The r=0 means that the vertex normal is used alone. As the neighbourhood ring size increases, calculation times fit an exponential trend with r^2=0.971.

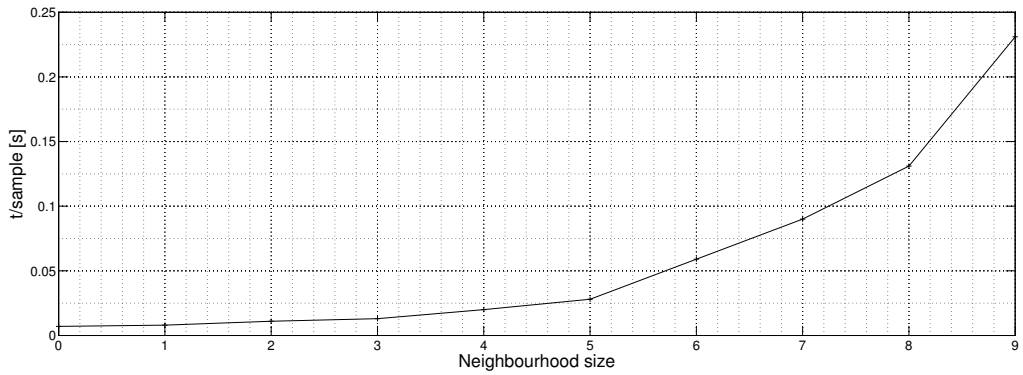


Figure 5.16: Feature calculation time, t [s] vs. neighbourhood ring size, r.

Table 5.6: Feature calculation time, t [s] vs. neighbourhood ring size, r.

Time,	**Neighbourhood ring size,** r									
t [s/*sample*]	0	1	2	3	4	5	6	7	8	9
Mean	0.007	0.008	0.011	0.013	0.020	0.028	0.059	0.090	0.131	0.231
σ	0.000	0.001	0.000	0.000	0.000	0.000	0.001	0.002	0.002	0.003
Min.	0.007	0.008	0.010	0.013	0.020	0.027	0.058	0.089	0.130	0.229
Max.	0.007	0.009	0.011	0.014	0.020	0.029	0.060	0.094	0.132	0.233
/1000samples	7.218	8.332	10.61	13.302	20.082	27.84	58.788	89.652	130.915	231.25

5.3.5 Methodology: feature sensitivity analysis

Given the feature vector definition of $\mathbf{X}\{z,\mathbf{n},\mathbf{n}\sigma^{1-5},\mathbf{T}\}$, each of the 22 input components has a different significance, importance, or sensitivity to the output. Whilst this varies between problems and geometry, a measure of importance can be calculated during generation of the reduced-order model based solely on the training data set. This is not possible with the ANN since, although the weights in the hidden layer neurons relate to the strength of each input, their direct relationship is intractable.

The random forest method (Breiman, 2001), specifically the *TreeBagger* (Matlab, n.d.) algorithm, intrinsically calculates the *OOBPermutedVarDeltaError*, defined as: *'a measure of importance for each predictor variable (feature). For any variable, the measure is the increase in prediction error if the values of that variable are permuted across the out-of-bag observations. This measure is computed for every tree, then averaged over the entire ensemble and divided by the standard deviation over the entire ensemble.'*

This method and algorithm have been used by Beaumont et al. (2014) for assessing the significance of features in an astrophysics image classification problem. During training, the RF calculates the importance of each component of the feature vector; and so components appearing often or closer to the root of the tree are deemed more important. The importance of each feature component is

more meaningful in a relative than an absolute sense, since features with higher importance have a greater impact on the structure of each decision tree.

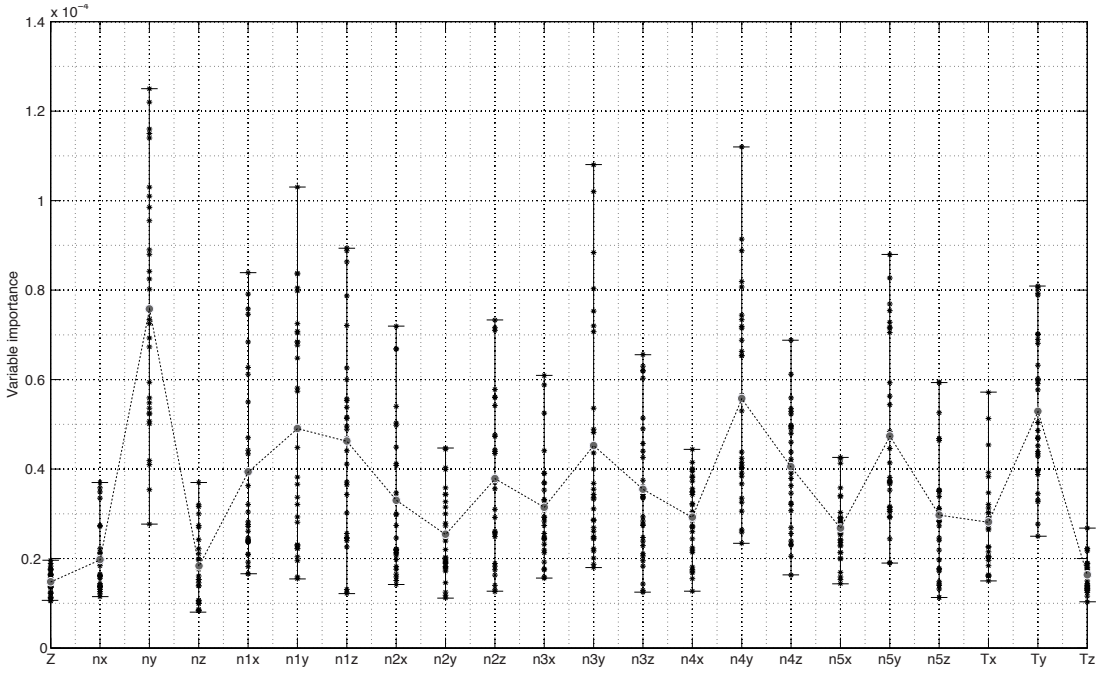


Figure 5.17: Feature importance for $\mathbf{X}\{z,\mathbf{n},\mathbf{n}\sigma^{1-5},\mathbf{T}\}$.

The training data from §6.2 is used since it has the broadest geometric complexity, being from the procedural tall building model, and of the largest size. A set size of 10000 randomly sampled from the full set is used; 10 trees for the *TreeBagger* algorithm; and the process is re-run 30 times to take the mean and range.

The first observation (Figure 5.17) is that primarily the y, and secondarily the x, components are typically always the most significant part of vector. The x (across-flow) and the y (stream-wise) direction components determine whether the point is facing into, perpendicular to, or away from the flow; which in turn is the primary indicator of a positive or negative pressure. Also, note that the variability or distribution increases with feature importance: the standard deviation σ and the mean have an r^2=0.795.

The variation of sample-based prediction error is related to the feature calculation times given in Table 5.6 and Figure 5.16. As more neighbourhoods rings are included, from $\mathbf{n}$ through to $\mathbf{n}\sigma^5$, the calculation time increases, but the prediction error decreases. However, improvements in accuracy with neighbourhood ring size, r, and calculation time, t, reduce as r increases (Figure 5.18).

Therefore, since t increases exponentially with r and improvements reduce with r, a compromise between time and accuracy is taken at r=5. Subsequently, the feature vector $\{Z,\mathbf{n},\mathbf{n}\sigma^{1-5},\mathbf{T}\}$ is used for the majority of this chapter unless otherwise stated.

Table 5.7: Feature calculation time, t [s], vs. neighbourhood ring size, r.

	Neighbourhood ring size, r					
	0	1	2	3	4	5
	Z,**n**,**T**	Z,**n**,**n**σ^1,**T**	Z,**n**,**n**σ^{1-2},**T**	Z,**n**,**n**σ^{1-3},**T**	Z,**n**,**n**σ^{1-4},**T**	Z,**n**,**n**σ^{1-5},**T**
Time, t [s/sample]	0.007	0.008	0.011	0.013	0.020	0.028
Error [%]						
Mean absolute	3.427	3.138	2.868	2.844	2.827	2.756
σ *absolute*	4.518	4.105	3.584	3.537	3.445	3.393
Min.	-38.005	-46.533	-45.258	-45.560	-53.178	-47.827
Max.	63.099	64.843	58.227	60.289	62.916	60.078

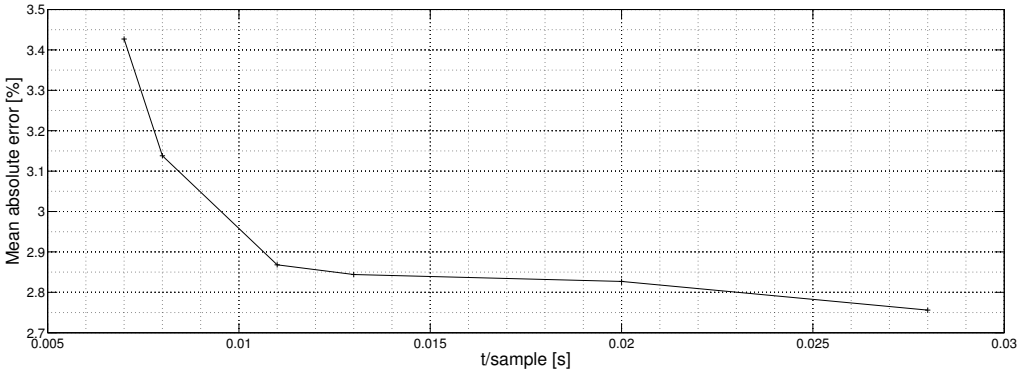
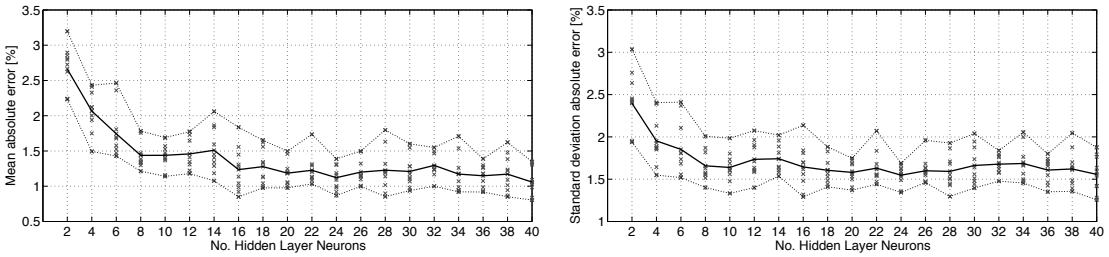


Figure 5.18: Feature calculation time, t [s], vs. prediction error [%].

5.3.6 Methodology: hidden layer sensitivity analysis

As introduced in §3.3.4, the ANN structure consists of a system of interconnected neurons, typically in at least three layers (input:hidden:output), which can be adjusted to map a set of inputs to outputs. Each input neuron is connected to every hidden layer neuron, and every hidden layer neuron to the output neuron (response) (Figure 3.18). Typically here, the ANN structure X:H:Y is 22:20:1, i.e. 22 input neurons, 20 hidden layer neurons, and 1 output.

The number of neurons in the hidden layer, H=20, was derived from a sensitivity analysis of H vs. sample-based error for the cuboid orientation case (§5.3.8). H was increased from 2 to 40 at intervals of 2; the ANN was re-trained 10-times for each H with a random training set and the average taken (Figure 5.19).



(a) Mean absolute [%] (b) Standard deviation absolute [%]

Figure 5.19: Hidden layer sensitivity analysis: H vs. sample-based error convergence.

There is no universally accepted method for prescribing the number of hidden layers or the number of neurons in those layers. General rules-of-thumb exist which suggest a value of H somewhere between the number of inputs and outputs (around H=12). Having run the sensitivity analysis on this case, this is in fact not a bad first guess for initial convergence. By H=20 however the sample-based errors have predominantly converged to a stable value of 1.183% mean absolute error, 1.614% standard deviation absolute error.

5.3.7 Methodology: ANN generalisability

To test the generalisability of the artificial neural network (ANN), prediction errors [%] are compared against those of the random forest (RF). Interpolation of surface pressure on a cuboid under orientation, as in the following section, is used as an example case. For this case: the training set size n=10000 taken from 15° increments, i.e. {0,15,30,45,60,75°}; no. hidden layer neurons H=20; no. random forest trees = 10. Figure 5.20 (Table 5.8) gives the model orientation against model-based mean absolute prediction errors [%] for both the ANN and RF.

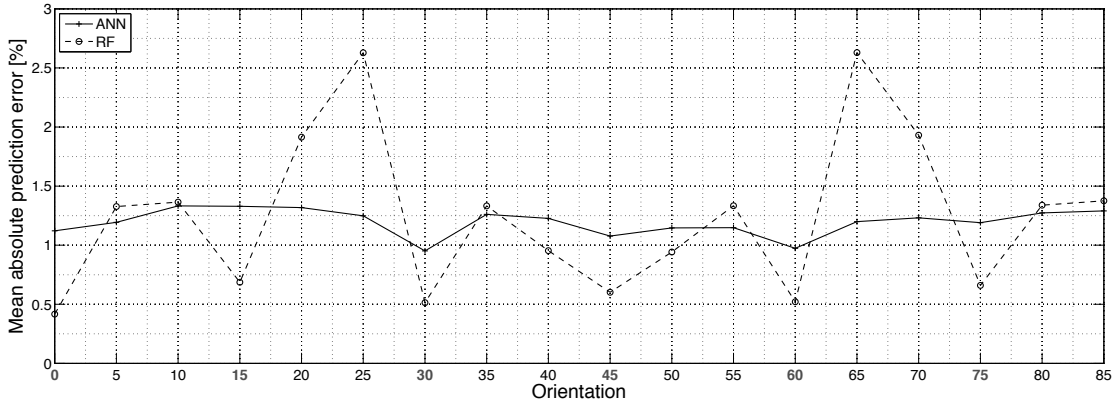


Figure 5.20: Cuboid orientation model-based interpolation: ANN vs. RF.

Table 5.8: Orientation: model-based ROM time vs. errors [%].

	ANN Error, δ [%]				**RF Error, δ [%]**			
Orientation	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$
(0)	-20.942	16.217	1.122	1.463	-11.673	6.541	0.417	0.681
5	-19.433	19.250	1.193	1.538	-14.631	12.873	1.327	1.407
10	-19.627	31.079	1.332	1.639	-10.691	26.897	1.365	1.633
(15)	-16.003	25.910	1.329	1.715	-9.112	25.691	0.686	1.229
20	-17.262	20.887	1.318	1.677	-16.334	22.989	1.915	2.830
25	-16.233	27.493	1.248	1.583	-21.042	31.851	2.628	4.265
(30)	-16.161	35.882	0.952	1.512	-15.266	38.817	0.514	1.062
35	-17.723	37.246	1.260	1.857	-12.849	36.207	1.332	1.905
40	-20.898	34.469	1.226	2.000	-11.745	41.851	0.954	1.565
(45)	-28.924	41.643	1.077	2.216	-11.444	40.327	0.602	1.460
50	-23.230	36.936	1.146	1.939	-10.716	39.665	0.942	1.580
55	-27.983	38.199	1.148	1.726	-13.165	41.442	1.334	1.948
(60)	-15.687	34.305	0.973	1.438	-13.500	44.706	0.521	1.051
65	-12.673	28.468	1.199	1.595	-20.736	31.387	2.630	4.209
70	-15.041	23.878	1.232	1.647	-16.885	22.200	1.931	2.889
(75)	-14.718	37.494	1.190	1.645	-10.181	27.977	0.659	1.166
80	-20.342	34.747	1.273	1.639	-13.044	23.453	1.338	1.593
85	-23.146	23.626	1.290	1.541	-13.544	10.092	1.376	1.387

It is clear that the RF has lower prediction errors on the training cases, as would be expected, however on new unseen orientations (5,10,20,25°...) the errors and the variability are much greater. The ANN on the other hand has a relatively consistent prediction error against orientation, demonstrating its ability to generalise or interpolate better than the RF. Therefore, the ANN will be used subsequently for all studies.

5.3.8 Cuboid orientation

The first study aims to test the reduced-order model on the simple rotation of a cuboid with the interpolation of surface pressure data. The cuboid has W:D:H=10:10:50m and is rotated around the Z-axis between 0° and 85° at 5° increments. *CFX* RANS was used to evaluate the resulting 18 models, and the input feature vector used here is:

$$f^{ANN} : (Z, \mathbf{n}_{x,y,z}, \mathbf{n}\sigma^{1-5}_{x,y,z}, \mathbf{T}_{x,y,z}) \rightarrow P \qquad \mathbf{S}_{(22+1,n)} \qquad \mathbf{T}_{(22,m)} \tag{5.14}$$

The general idea is depicted in Figure 5.21; of using fixed orientation intervals for training data and intermediate orientations for test data. Therefore, surface pressure is essentially interpolated between orientations.

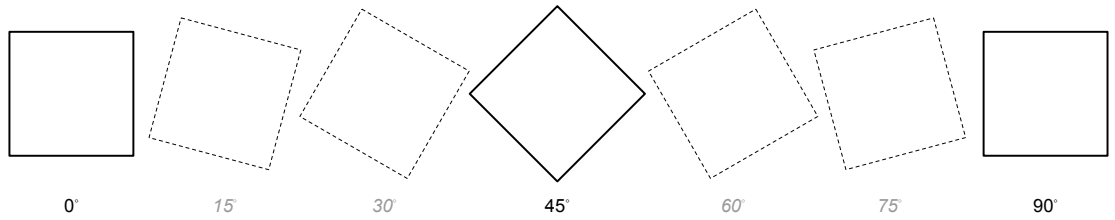
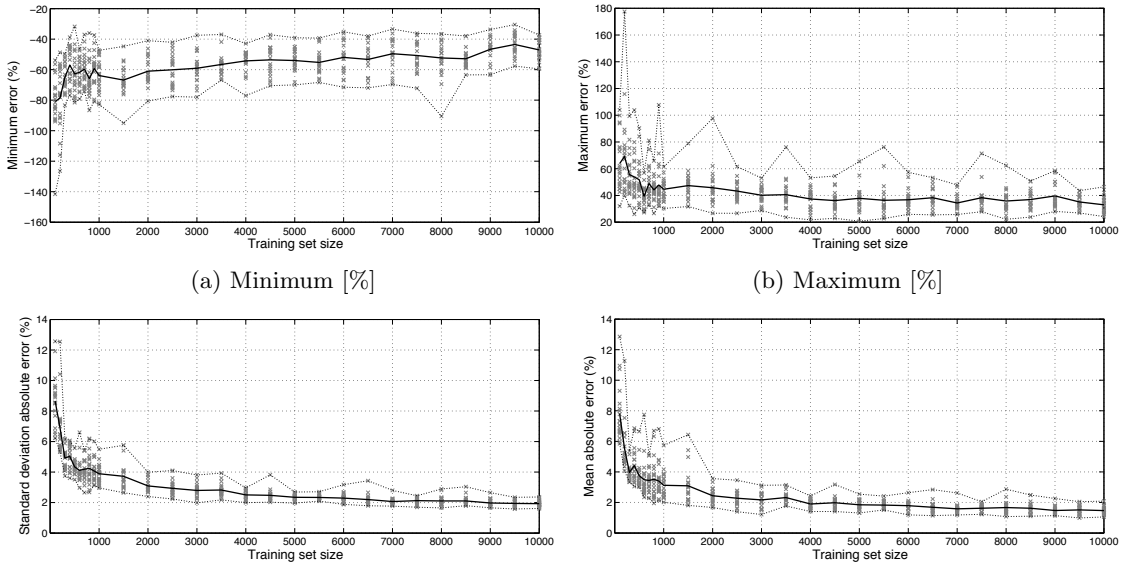


Figure 5.21: Cuboid orientation model-based interpolation principle.

In Figure 5.21, the training set consists of {0,45,90°} models, and the test set of {15,30,60,75°} models. In this study, a number of different increment sizes are tested for comparison; although for the final model-based tests are taken from the 15° training increment with 5° intermediate test intervals. Note, that for this study the feature vector could be replaced with a higher-level representation such as $\{x, y, z, r\}$, where each vertex's coordinates and the model rotation are used as inputs. This is possible since each model is nearly identical, except for its global orientation; however, for subsequent tests this is not feasible.

Results: sample-based

There are a total of D=334339 samples available for training and testing, and so after selecting the training set size, the test set consists of the remaining available samples. Figure 5.22 shows the error convergence as the training set size is increased. Each shows the training convergence from 20 individual runs (grey points), the error range (dotted lines), and the mean (solid line).



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 5.22: Cuboid orientation: sample-based training error convergence.

The errors given are at the converged training set size (Figure 5.22) of n=10000. The sample-based errors are therefore: min. = -47.090%, max. = 32.981%, mean abs. = 1.471%, and σ abs. = 1.929%.

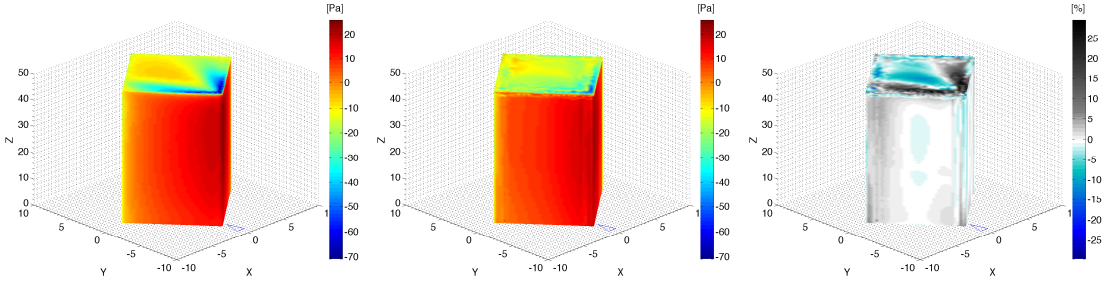
Results: model-based

In the second assessment, instead of including all models at 5° increments in the training set (from which features are randomly sampled), in the model-based test only discrete orientation intervals are used. Training set samples are taken from the 15° orientation intervals, leaving the remaining intervals free for independent model tests. Prediction errors and times are given in Table 5.9.

Figure 5.23 visualises the orientation with the minimum overall error, 40°; and Figure 5.24 with the maximum overall error, 10°. Model-based test set predictions can be visualised in their original shape, showing the simulated (left, Y $[Pa]$), predicted (centre, Y' $[Pa]$), and the difference or error between the two (right, δ [%]). Since the pressure can be either positive or negative, over-prediction is shown by grey-black and under-prediction by green-blue.

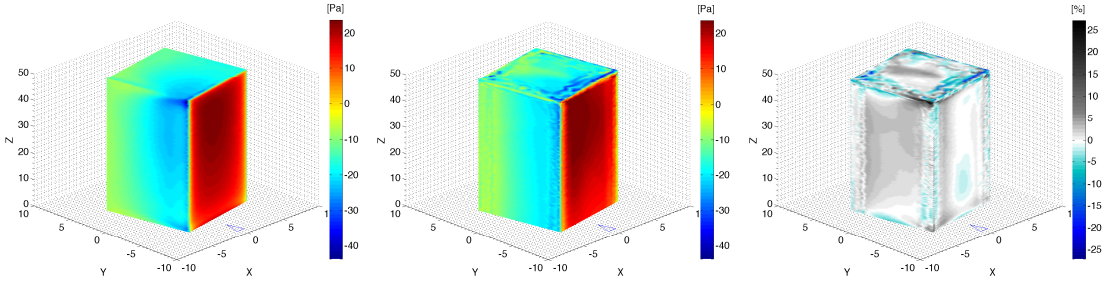
Table 5.9: Orientation: model-based ROM time vs. errors [%].

		Error, δ [%]				**Test time, t [s]**		
Orientation	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
(0)	17663	-	-	-	-	-	-	-
5	16329	-28.925	15.251	1.198	1.423	454.607	0.0655	454.673
10	17354	-17.736	22.166	1.217	1.466	483.144	0.0615	483.205
(15)	17493	-	-	-	-	-	-	-
20	18353	-17.432	20.370	1.053	1.333	510.957	0.0623	511.019
25	18966	-16.108	20.281	1.047	1.299	528.023	0.0615	528.084
(30)	19612	-	-	-	-	-	-	-
35	19902	-14.273	36.288	0.943	1.451	554.081	0.0656	554.147
40	20107	-15.169	37.430	0.865	1.555	559.789	0.0608	559.850
(45)	20418	-	-	-	-	-	-	-
50	20122	-14.624	38.790	0.871	1.638	560.206	0.0610	560.267
55	19912	-16.117	36.593	0.917	1.436	554.360	0.0625	554.422
(60)	19587	-	-	-	-	-	-	-
65	18946	-14.548	19.698	1.137	1.429	527.466	0.0616	527.528
70	18302	-16.366	21.566	1.108	1.449	509.537	0.0642	509.601
(75)	17557	-	-	-	-	-	-	-
80	17364	-27.819	27.931	1.154	1.492	483.422	0.0639	483.486
85	16352	-20.512	16.754	1.090	1.375	455.248	0.0625	455.310



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.23: Cuboid orientation: min. $\overline{\lvert\delta\rvert}$ test model = 40°.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.24: Cuboid orientation: max. $\overline{\lvert\delta\rvert}$ test model = 10°.

In the first case, highest errors in both over- and under-prediction are found on the top face of the cuboid; whilst in the second, highest errors are generally located around the edges and corners.

Results: model-based orientation interpolation intervals

The interval between models used for the training set is varied at 10, 15, 30, and 45° to see its effect on the interpolation error (Figure 5.25). A constant training set size was used (10000), however

the source of the data changes for each case.

Table 5.10: Orientation interpolation: model-based ROM errors [%].

		Error, $\overline{\lvert\delta\rvert}$ [%]			
Orientation	m	**10° inc.**	**15° inc.**	**30° inc.**	**45° inc.**
0	17663	-	-	-	-
5	16329	1.702	1.198	1.807	2.283
10	17354	-	1.217	2.963	3.823
15	17493	1.874	-	3.163	4.395
20	18353	-	1.053	2.212	3.807
25	18966	1.753	1.047	1.120	2.817
30	19612	-	-	-	2.072
35	19902	1.519	0.943	1.091	1.429
40	20107	-	0.865	1.423	0.899
45	20418	1.454	-	1.722	-
50	20122	-	0.871	1.445	0.999
55	19912	1.339	0.917	1.049	1.530
60	19587	-	-	-	2.303
65	18946	1.443	1.137	1.080	3.144
70	18302	-	1.108	2.065	4.174
75	17557	1.968	-	2.943	4.444
80	17364	-	1.154	2.759	3.571
85	16352	1.611	1.09	1.701	2.088

For 45 and 30° training intervals, the highest errors are generally at the furthest points from the training model orientations indicating poor generalisation or interpolation. However, for 10 and 15° training intervals, the errors are more uniform across the orientation range. In fact, 15° intervals has a lower overall error across the range than 10° intervals; although the difference is no more than 0.5%.

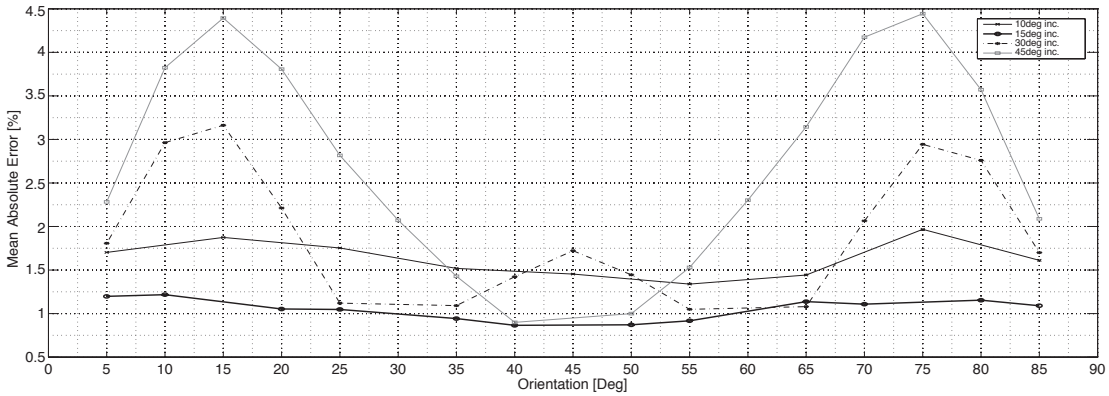


Figure 5.25: Cuboid orientation model-based interpolation: training increment variation.

5.3.9 Cuboid height

The second study tests the reduced-order model on a simple cuboid with varying height. The cuboid has dimensions W:D=10:10m, and its height varies between 10m and 100m at 5m increments. Each of the 19 models is evaluated with *CFX* RANS and the input feature vector (Figure 5.15) used

here is:

$$f^{ANN} : (Z, \mathbf{n}_{x,y,z}, \mathbf{n}\sigma^{1-5}_{x,y,z}, \mathbf{T}_{x,y,z}) \rightarrow P \qquad \mathbf{S}_{(22+1,n)} \qquad \mathbf{T}_{(22,m)} \tag{5.15}$$

The general idea is depicted in Figure 5.26; of using fixed height intervals for training data (solid lines) and intermediate heights for test data (dashed lines). Therefore, surface pressure is essentially interpolated between cuboids of different heights without any explicit higher level 'height' parameter.

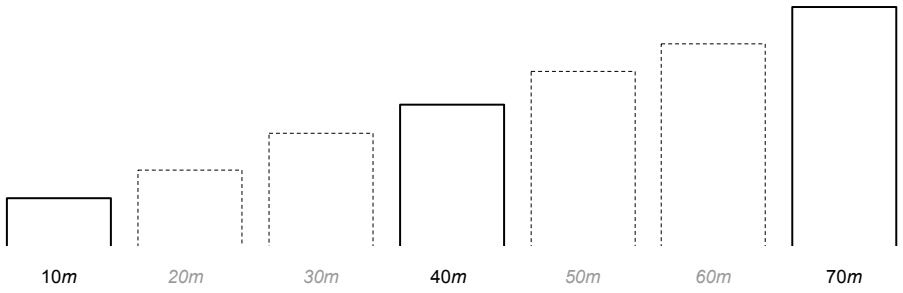
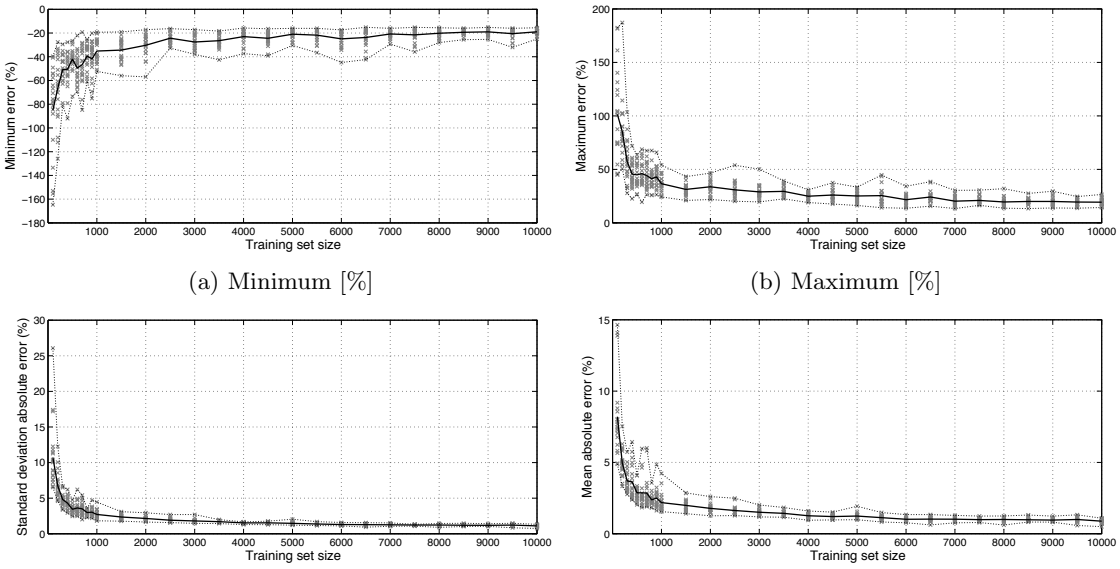


Figure 5.26: Cuboid height model-based interpolation principle.

Results: sample-based

There are a total of D=185155 samples available for training and testing, each with corresponding pairs of X and Y. As such, after selecting the training set size the test set consists of the remaining available samples. Figure 5.27 shows the ANN learning convergence as the training set size is increased.



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 5.27: Cuboid height: sample-based training error convergence.

Note that the maximum training set size does not exceed $1e^4$ since the error converges. Each shows

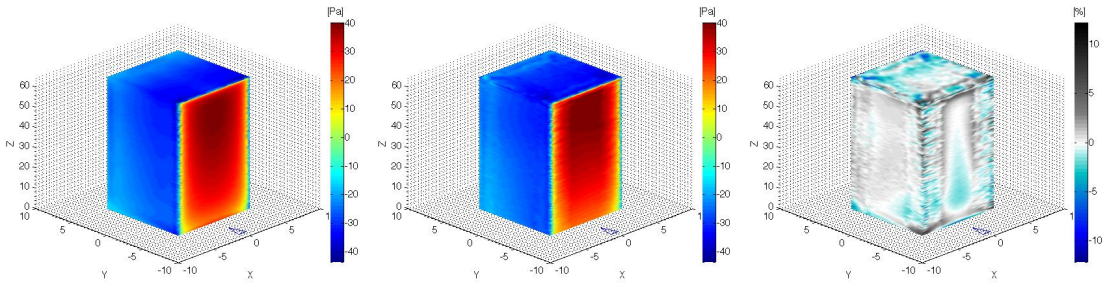
the training convergence from 20 individual runs (grey points), the error range (dotted lines), and the mean (solid line). The errors given are at the converged training set size (Figure 5.27) of n=10000. The sample-based errors are therefore: min. = -18.985%, max. = 19.307%, mean abs. = 0.878%, and σ abs. = 1.102%.

Results: model-based

In the second assessment, instead of including all models at $5m$ increments in the training set (from which features are randomly sampled), in the model-based test only discrete heights intervals are used. Training set samples are taken from the $15m$ height intervals, leaving the remaining intervals free for independent model tests. Prediction errors and times are given in Table 5.11.

Table 5.11: Height: model-based ROM time vs. errors [%].

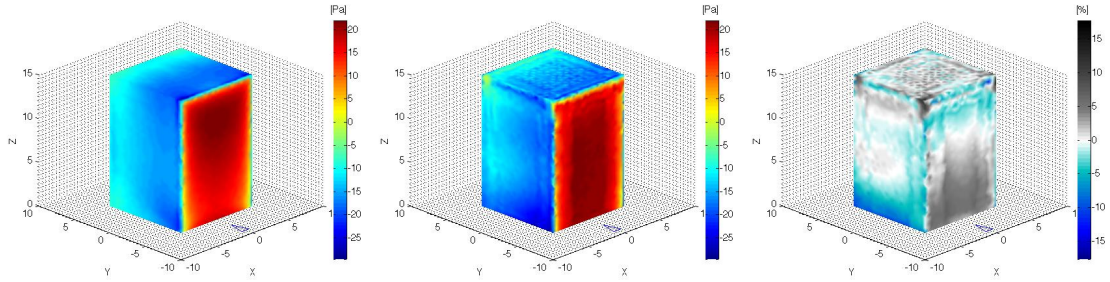
		Error, δ [%]				**Test time, t [s]**		
Height	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
(10)	2144	-	-	-	-	-	-	-
15	3156	-12.066	13.619	1.632	1.950	87.863	0.0206	87.884
20	3885	-10.113	17.109	1.374	1.741	108.158	0.0250	108.183
(25)	4682	-	-	-	-	-	-	-
30	5596	-8.401	13.426	0.991	1.264	155.793	0.0289	155.822
35	6328	-7.784	9.250	0.876	1.055	176.172	0.0315	176.203
(40)	7280	-	-	-	-	-	-	-
45	8011	-7.266	11.855	0.764	0.839	223.026	0.0368	223.063
50	8909	-6.938	13.422	0.731	0.833	248.027	0.0392	248.066
(55)	9675	-	-	-	-	-	-	-
60	10626	-7.979	13.131	0.673	0.762	295.828	0.0461	295.874
65	11351	-8.480	7.540	0.629	0.716	316.012	0.0487	316.061
(70)	12355	-	-	-	-	-	-	-
75	13006	-11.144	11.955	0.635	0.748	362.087	0.0515	362.139
80	13964	-12.088	12.008	0.674	0.820	388.758	0.0513	388.809
(85)	14689	-	-	-	-	-	-	-
90	15719	-16.269	12.529	0.705	0.886	437.617	0.0612	437.678
95	16379	-10.911	13.243	0.747	0.918	455.991	0.0634	456.055
(100)	17400	-	-	-	-	-	-	-



(a) Y [Pa] (b) Y' [Pa] (c) δ [%]

Figure 5.28: Cuboid height: min. $\overline{\lvert\delta\rvert}$ test model = $65m$.

Prediction errors tend to decrease from a maximum at $15m$ (Figure 5.29) as height increases, to a minimum at $65m$ (Figure 5.28). The reasons for the highest errors occurring at the lower heights is potentially due to boundary layer turbulence from ground surface roughness.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.29: Cuboid height: max. $\overline{|\delta|}$ test model = $15m$.

Note that for all the studies a power-law profile is used on the vertical wind speed distribution at the inlet. For the previous orientation and the following topology studies, the height of all the models remained constant; however in this case the heights vary from 10 to $100m$. Therefore, since wind speeds increase with height, the pressure range is also increasing. This can be seen when comparing the $15m$ model with a pressure range of -29 to $21Pa$, and the $65m$ with a range of -42 to $40Pa$.

Results: model-based height interpolation intervals

For 10, 15, and $30m$ training intervals, generalisation or interpolation is successful; however, at $45m$ intervals the errors are again greatest at the furthest points from the training data. It is clear that the $15m$ intervals gives the lowest prediction errors (Figure 5.30 and Table 5.12).

Table 5.12: Height interpolation: model-based ROM errors [%].

		Error, $\overline{\lvert\delta\rvert}$ [%]			
Height	m	**10m inc.**	**15m inc.**	**30m inc.**	**45m inc.**
10	2144	-	-	-	-
15	3156	1.715	1.632	2.273	2.383
20	3885	-	1.374	2.099	2.570
25	4682	1.246	-	1.839	2.643
30	5596	-	0.991	1.580	2.661
35	6328	0.965	0.876	1.286	2.466
40	7280	-	-	-	2.168
45	8011	0.785	0.764	1.018	1.778
50	8909	-	0.731	0.988	1.361
55	9675	0.695	-	0.915	-
60	10626	-	0.673	0.866	1.287
65	11351	0.652	0.629	0.831	1.604
70	12355	-	-	-	1.867
75	13006	0.674	0.635	0.844	1.885
80	13964	-	0.674	0.886	1.927
85	14689	0.761	-	0.881	1.799
90	15719	-	0.705	0.925	1.600
95	16379	0.874	0.747	0.953	1.199
100	17400	-	-	-	-

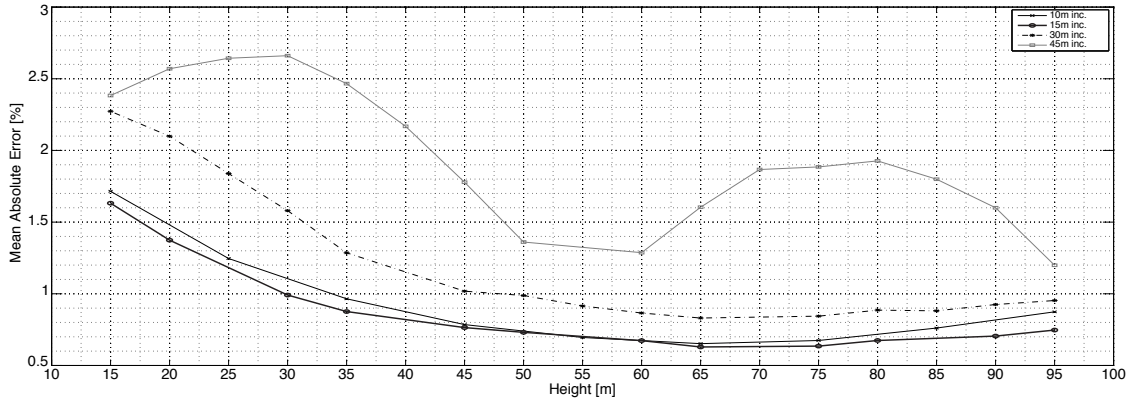


Figure 5.30: Cuboid height model-based interpolation: training increment variation.

5.3.10 Topology

The third test is for varying topology, or a prism with a variable number of edges. The prism has maximum dimensions W:D:H=10:10:50m, with 3 to 9 plus 0 (cylinder) edges. Each of the 9 models is evaluated with *CFX* RANS. The input feature vector (Figure 5.15) used here is:

$$f^{ANN} : (Z, \mathbf{n}_{x,y,z}, \mathbf{n}\sigma^{1-5}_{x,y,z}, \mathbf{T}_{x,y,z}) \rightarrow P \qquad \mathbf{S}_{(22+1,n)} \qquad \mathbf{T}_{(22,m)} \tag{5.16}$$

The general idea is depicted in Figure 5.31; of using fixed intervals in the number of edges for training data and intermediate models for test data. Therefore, surface pressure is essentially interpolated between prisms with different numbers of edges.

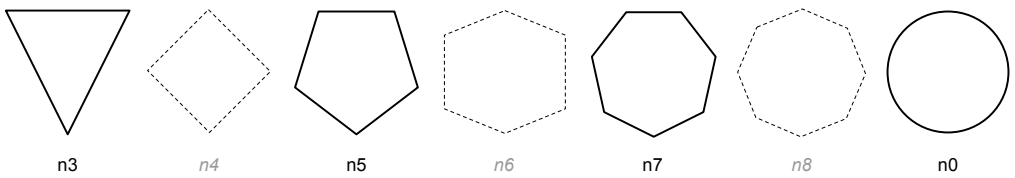


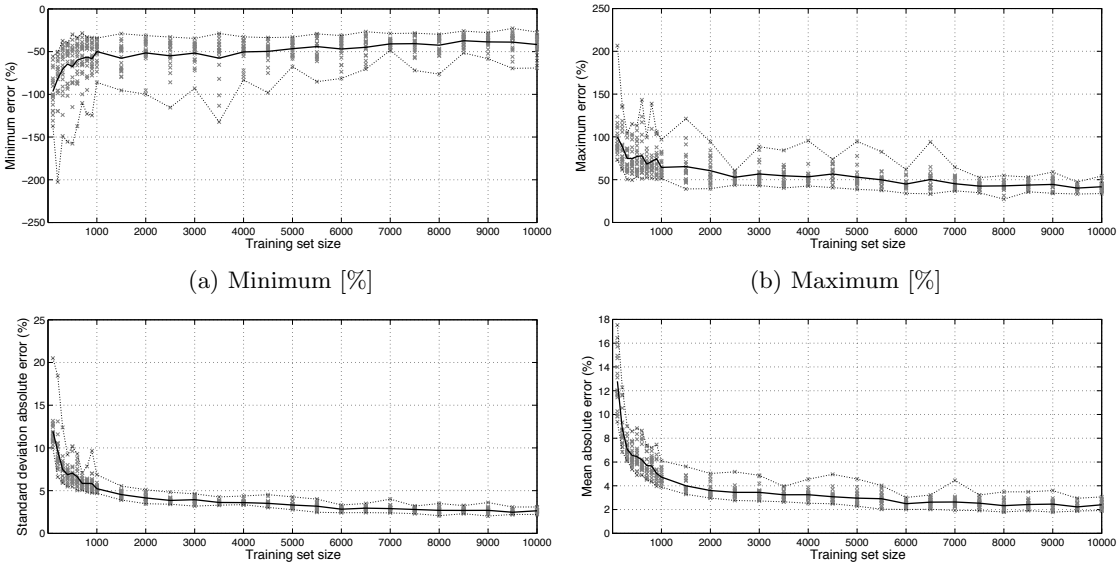
Figure 5.31: Topology model-based interpolation principle.

This study is more complex than the previous two since the topology changes affect the flow behaviour more significantly. This can be seen in the very different pressure distributions on the model-based plots in Figures 5.33 to 5.36.

Results: sample-based

There are a total of D=166285 samples available for training and testing, each with corresponding pairs of X and Y. As such, after selecting the training set size the test set consists of the remaining available samples. Figure 5.32 shows the ANN training convergence as the training set size is increased. Each shows the training convergence from 20 individual runs (grey points), the error

range (dotted lines), and the mean (solid line).



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 5.32: Topology: sample-based training error convergence.

The errors given are at the converged training set size (Figure 5.32) of n=10000. The sample-based errors are therefore: min. = -41.692%, max. = 41.738%, mean abs. = 2.398%, and σ abs. = 2.644%.

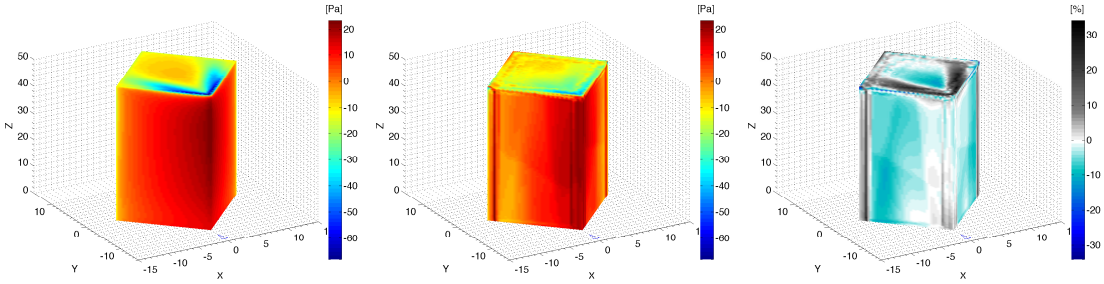
Results: model-based

The model-based test results are given below in Table 5.13, and visually in Figures 5.33 to 5.36. Prediction errors are higher than from the previous orientation and height interpolation studies, although still relatively low.

Table 5.13: Topology: model-based ROM time vs. errors [%].

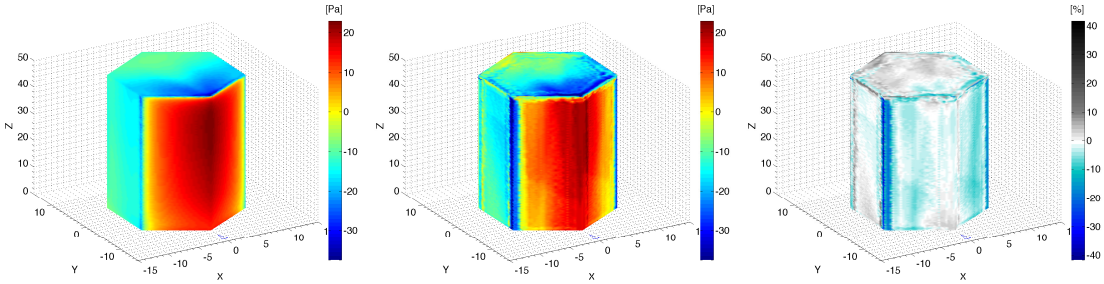
		Error, δ [%]				**Test time, t [s]**		
No. Edges	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
(3)	14825	-	-	-	-	-	-	-
4	18805	-19.014	32.532	4.941	4.806	523.531	0.0618	523.593
(5)	18318	-	-	-	-	-	-	-
6	19075	-52.507	37.713	6.475	6.266	531.048	0.0624	531.110
(7)	18792	-	-	-	-	-	-	-
8	18681	-30.023	19.520	3.448	3.893	520.079	0.0653	520.144
(9)	18852	-	-	-	-	-	-	-
10	19245	-34.316	33.186	4.455	4.434	535.781	0.0614	535.842
(0)	19692	-	-	-	-	-	-	-

In general, the fidelity of the simulated pressure distributions are conveyed by the predictions. There are however regions where local distributions are not captured accurately; for instance, the edge regions on the top face of $n4$ and the side faces of $n6$. These are regions where turbulence is greatest and the pressure conditions are idiosyncratic to the specific geometry. It is therefore difficult to interpolate or generalise from other geometries and make accurate local predictions.



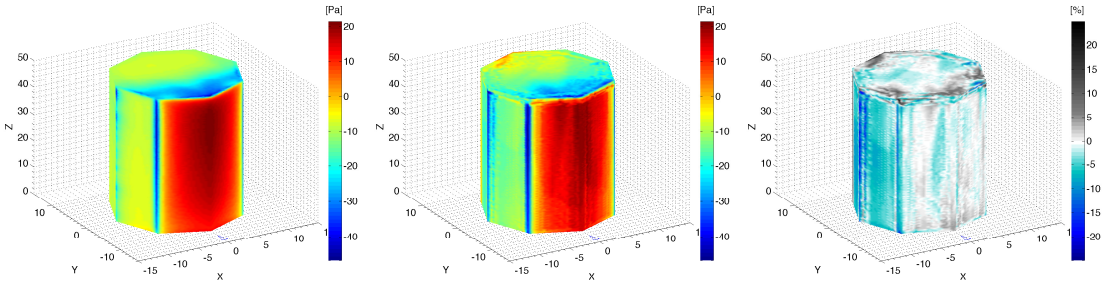
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.33: Topology: test model = $n4$.



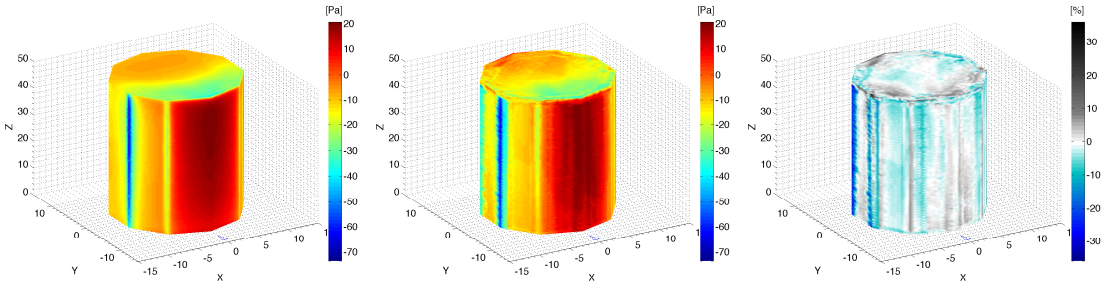
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.34: Topology: test model = $n6$.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.35: Topology: test model = $n8$.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.36: Topology: test model = $n10$.

A challenge with this topology study is that there is no way to decrease the training interval between models, i.e. a model with 3.5 edges does not exist. This problem does not occur with the previous studies where more intervals can simply be used to improve the predictions. However,

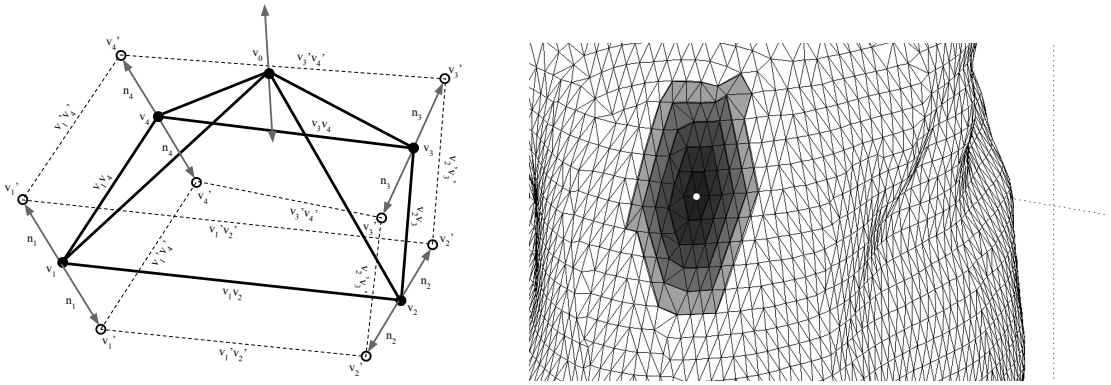
this specific topological problem is localised to this specific study and is overcome is subsequent sections, namely in the case of the procedural tall building model being used for predictions on real buildings.

5.3.11 Methodology: convex and concave curvature

In this case, the features have been updated to allow for positive and negative standard deviations of the neighbourhood curvature. The extension of the input feature standard deviation (Figure 5.15) to be positive for convex vertex neighbourhoods and negative for concave vertex neighbourhoods allows for greater accuracy and applicability to a broader set of forms. Initially, for simpler geometries this was not necessary; however for the complex forms of realistic buildings encountered in the following chapter, concave regions are far more common. The full calculation code is given in Appendix B.2, and shown in Figure 5.37, but the basic procedure is as follows:

for vertex v_i, its offset is the vertex plus normal $v'_i = v_i + n_i$
  if $(v'_1 ... v'_n > v_1 ... v_n)$ then $v_1 ... v_n$ is convex, σ^+
  if $(v'_1 ... v'_n < v_1 ... v_n)$ then $v_1 ... v_n$ is concave, σ^-
  else $v_1 ... v_n$ is planar, $\sigma = 0$

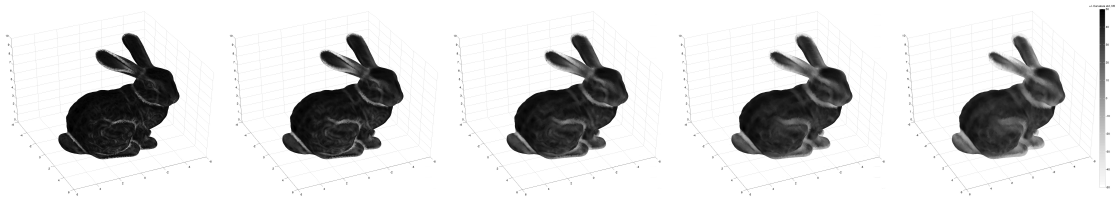
The use of various neighbourhood scales (rings one through five) gives the local curvature over a range of scales, as can be seen in Figure 5.38. In these images the white regions are concave and black are convex. Due to the complexity of the meshes here though certain points may in fact be convex in one axis and concave in another. For this reason the x, y, and z components are given for each scale. The images show the mean of all three components to give the primary indication of curvature direction.



(a) $N1$ convexity / concavity calculation. (b) Vertex neighbourhoods $N1 - 5$.

Figure 5.37: Local curvature analysis.

The convex-concave calculation is shown in Figure 5.38 on the Stanford Bunny (Turk & Levoy, 1994), a standard test model used in computer graphics. The model consists of 69451 triangular polygons, and shows the extension of the curvature neighbourhood from the vertices' first to fifth neighbourhood rings.



(a) $N1$-ring (b) $N2$-ring (c) $N3$-ring (d) $N4$-ring (e) $N5$-ring

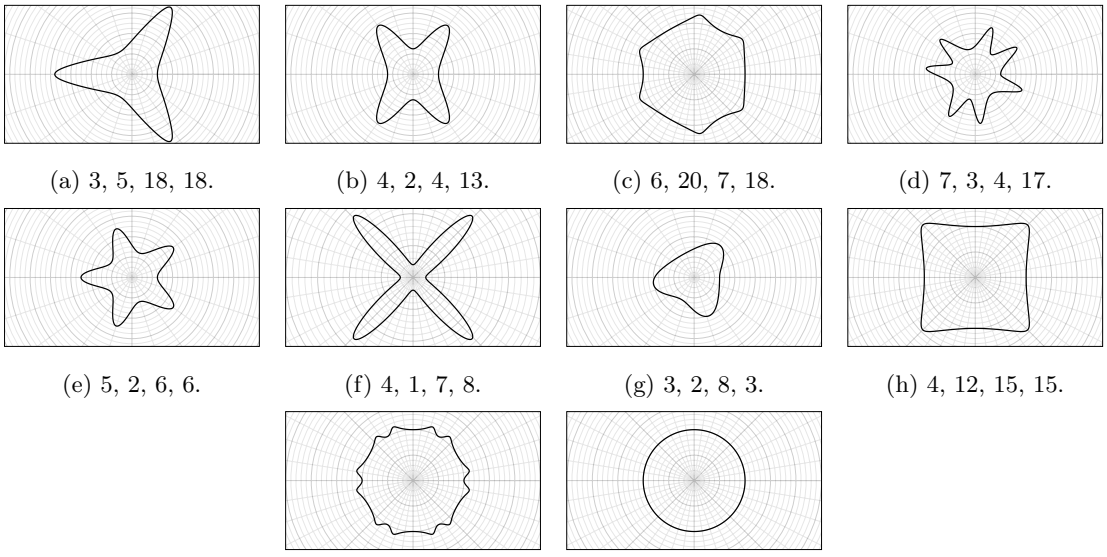
Figure 5.38: Convex-concave curvature analysis over $N1$ to $N5$ neighbourhoods.

5.3.12 Super-formula

For testing the convex-concave description a 'super-formula' shape definition is used. This is a generalised form of the super-ellipse and was first proposed by Gielis (2003).

$$r(\varphi) = \left[\left| \frac{cos(\frac{m\varphi}{4})}{a} \right|^{n_2} + \left| \frac{sin(\frac{m\varphi}{4})}{b} \right|^{n_3} \right]^{-\frac{1}{n_1}} \tag{5.17}$$

Where m is the rotational symmetry, φ the angle, and a and b the x and y radii. The following set of 10 possible plan forms are randomly selected for their variation. To create 3-d models they are simply extruded to a height of $50m$. An x and y radius, $a = b$, of $10m$ is used. In the following set, the four numbers are: m, n_1, n_2, and n_3. The set was selected at random.



(a) 3, 5, 18, 18. (b) 4, 2, 4, 13. (c) 6, 20, 7, 18. (d) 7, 3, 4, 17.

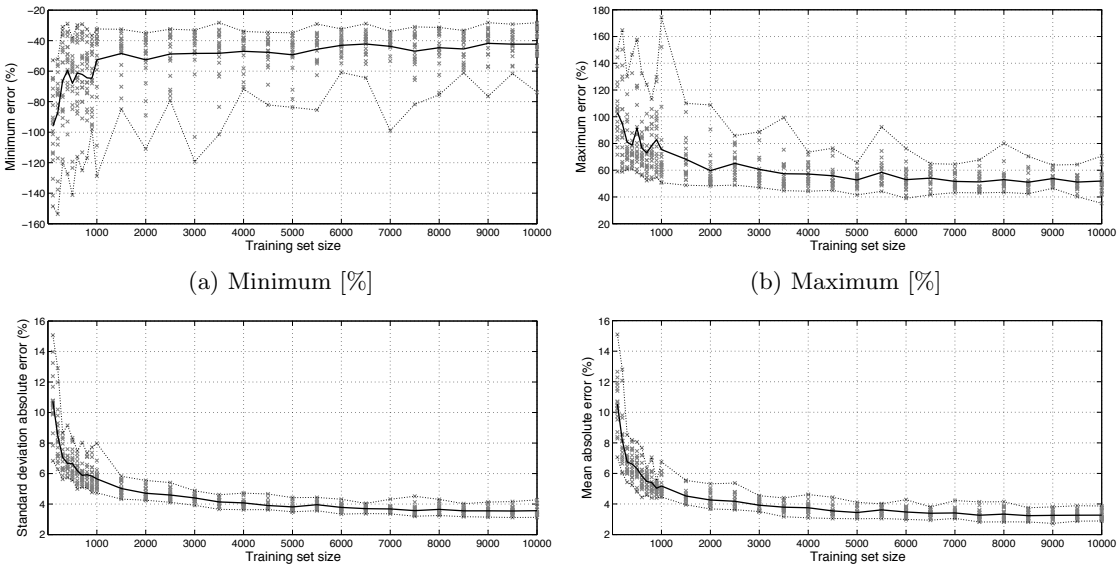
(e) 5, 2, 6, 6. (f) 4, 1, 7, 8. (g) 3, 2, 8, 3. (h) 4, 12, 15, 15.

(i) 12, 15, 20, 3. (j) 2, 2, 2, 2.

Figure 5.39: Super-formula training and test set plans.

Results: sample-based

There are a total of D=243440 samples available for training and testing, each with corresponding pairs of X and Y; after selecting the training set size the test set consists of the remaining available samples. Figure 5.40 shows the ANN convergence as the training set size is increased.



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 5.40: Super-formula: sample-based training error convergence.

The errors given are at the converged training set size (Figure 5.40) of n=10000. The sample-based errors are therefore: $\delta_{min.}$ = -42.284%, $\delta_{max.}$ = 51.890%, $\overline{|\delta|}$ = 3.269%, and $\sigma_{|\delta|}$ = 3.565%.

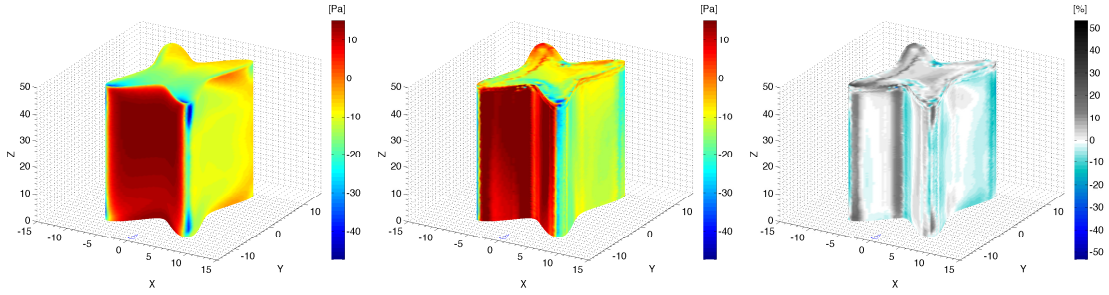
Results: model-based

The model-based test here differs from the previous cases as the model set are not a linear sequence, but rather a random set of 10 instances from the super-formula model. In this way, the test is more similar to the procedural training model methodology used in the following chapter.

Table 5.14: Super-formula: model-based ROM time vs. errors [%].

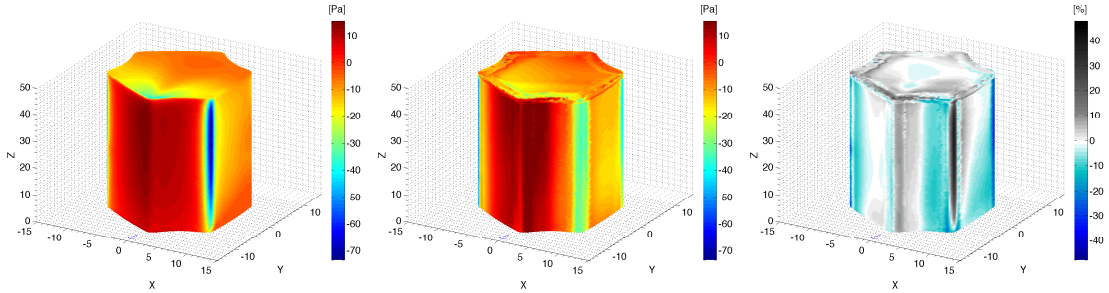
		Error, δ [%]				**Test time, t [s]**		
Super-formula	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
(3,5,18,18)	29888	-	-	-	-	-	-	-
4,2,4,13	22204	-41.481	30.215	5.872	6.296	618.159	0.0789	618.238
6,20,7,18	30021	-52.015	49.347	8.140	8.315	835.785	0.0920	835.877
(7,3,4,17)	28724	-	-	-	-	-	-	-
5,2,6,6	24054	-27.316	52.350	5.024	4.532	669.663	0.0816	669.745
4,1,7,8	31270	-33.097	46.344	6.768	5.893	870.557	0.0978	870.655
(3,2,8,3)	21970	-	-	-	-	-	-	-
4,12,15,15	18013	-40.048	33.989	5.950	5.279	501.482	0.0702	501.552
12,15,20,3	17637	-42.660	34.478	7.423	6.003	491.014	0.0699	491.084
(2,2,2,2)	19659	-	-	-	-	-	-	-

The model-based test results in Figures 5.41 to 5.46 again show that in general, the fidelity of the simulated pressure distributions are conveyed by the predictions. There are however regions where local distributions are not captured accurately.



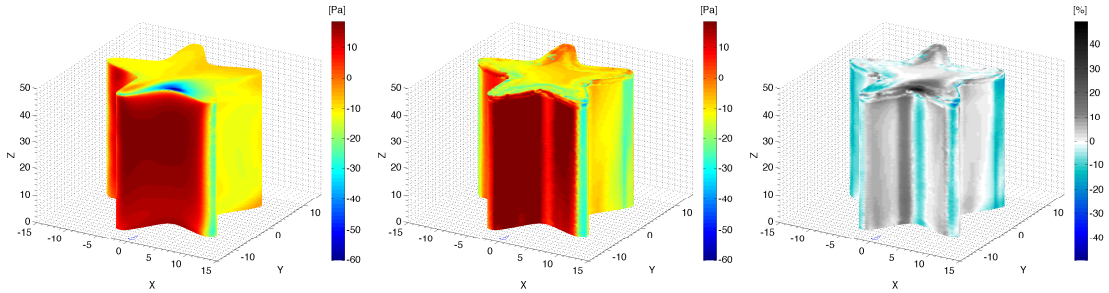
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.41: Super-formula: test model = 4,2,4,13.



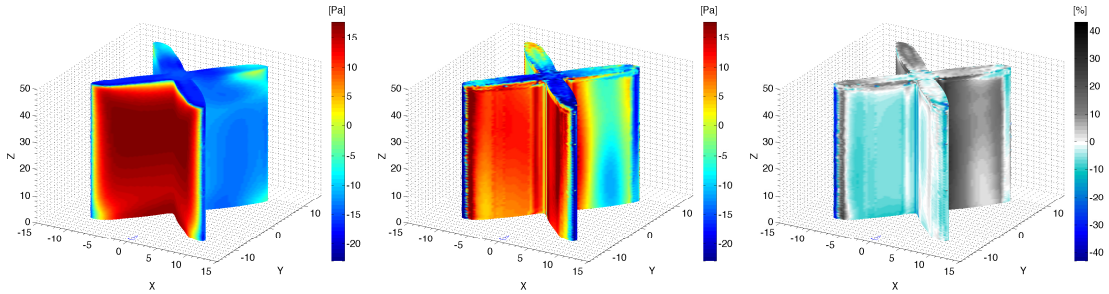
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.42: Super-formula: test model = 6,20,7,18.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

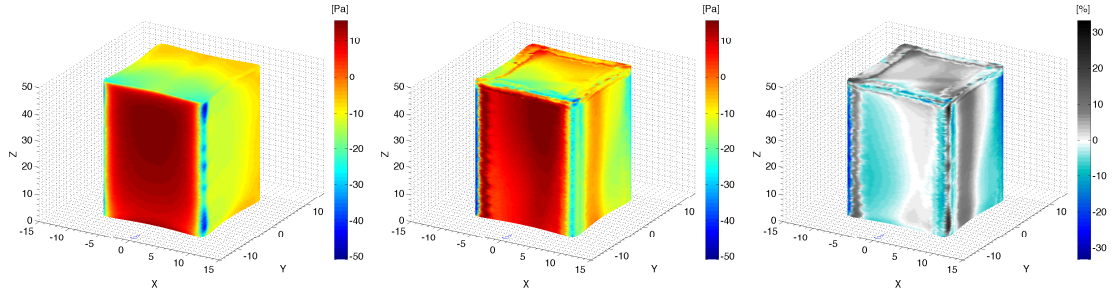
Figure 5.43: Super-formula: test model = 5,2,6,6.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

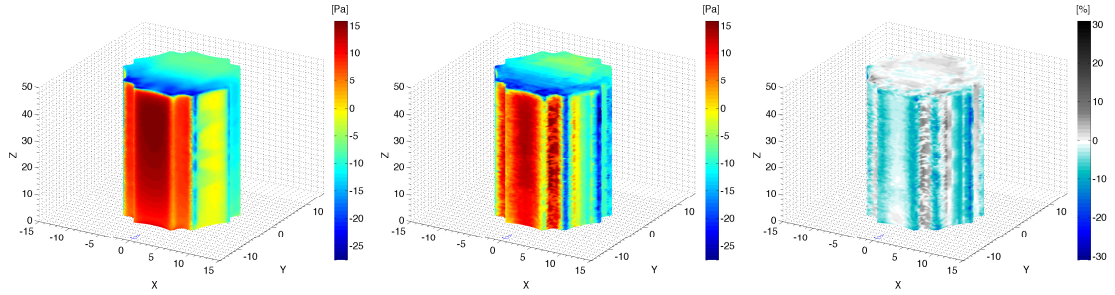
Figure 5.44: Super-formula: test model = 4,1,7,8.

The highest errors are primarily located on the edges of side faces and in deeply concave areas. These are regions where turbulence is greatest and the pressure conditions are idiosyncratic to the specific geometry and it is therefore difficult to interpolate or generalise from other geometries and



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.45: Super-formula: test model = 4,12,15,15.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.46: Super-formula: test model = 12,15,20,3.

therefore make accurate local predictions. Or similarly, for deeply concave regions like on Figure 5.44 there may just be limited training data.

5.4 Interference

In the previous test cases, predictions were made in isolation without any surrounding context or interference. This section develops the reduced-order model definition further to consider flows in which the design may have surrounding buildings which alter the flow conditions. In reality this is a common occurrence and is therefore crucial to the applicability of the method.

In the review, interference was introduced in the context of wind engineering (§3.2.2) and attempts at generalisation from wind-tunnel and CFD studies (§3.3.6). The following limitations of existing literature were noted: firstly, all of the studies use similar cuboids since form is not considered; secondly, basic configurations of typically only two or three buildings are included due to the combinatorial approach; and thirdly, a lack of output data since the interference factor (IF) can only provide a factor indicating the change over the whole model. The esoteric nature of each situation is indeed a challenge to approximation, however an alternative solution is proposed and tested here.

The approach here is to combine: i) a large-scale CFD simulation of an urban context, the *obstruction model (OM)*, which remains static throughout the design process and can therefore be

simulated only once; and ii) a small-scale ROM prediction of an isolated tall building, the *principal model (PM)*, which can be iteratively changed through generative design.

Considerable time can be saved if it can be demonstrated that a CFD simulation can effectively be used for boundary conditions to a ROM. The clear advantage is that the full OM does not need to be re-run with every change of the PM.

5.4.1 Methodology

Previously the inlet boundary conditions have followed the basic vertical power-law profile distribution. This has meant it was sufficient to include the vertical position of a vertex as a feature, corresponding to its relative position on the profile. Whilst this boundary condition is still used in the OM simulation, the addition of surrounding buildings alters the upstream wind speed that affects the PM surface pressure. This necessitates inclusion of the local flow field wind speed in the input feature.

In the test case (the OM simulation), the wind speed is applied at an upstream inlet with a reference speed (v_r) of $10m \cdot s^{-1}$ at a reference height (z_r) of $10m$. In the training models a constant vertical wind profile is used, albeit with varying speeds, so as to generate a range of upstream wind speeds across the simulated training set for every vertex, i.e. $v_x = v_r$.

For a training set, $\mathbf{S}$, consisting of vertex feature vectors and simulated pressure extracted from the CFD, the ANN approximates the function $f^{ANN} : \mathbf{X} \rightarrow P$ where $\mathbf{X}$ is the vertex feature vector and P is the vertex pressure. $\mathbf{X}$ is defined as:

$$\mathbf{X}\{v, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{T}\} \tag{5.18}$$

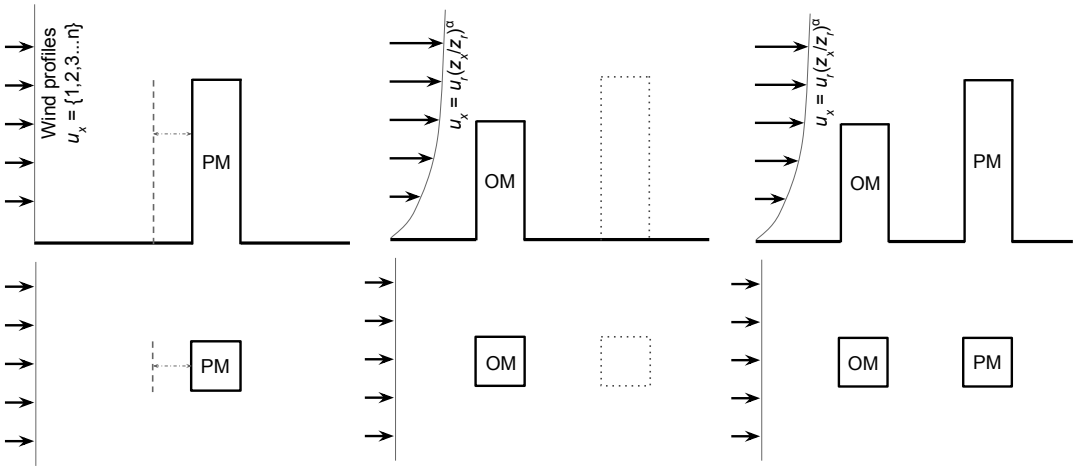
where $\mathbf{n}_{x,y,z}$ are the vertex normal components; $\mathbf{n}\sigma^{1-5}_{x,y,z}$ are the vertex-ring (one through five) neighbourhood curvature (non-absolute) standard deviation components; and $\mathbf{T}_{x,y,z}$ are the normalised relative vertex position within the model limits. For training, v_S is simply the inlet wind speed of the training simulation which is constant with height, i.e. no profile. For testing however, this is replaced with v_T, the wind speed at the vertex's transformed position in the OM field.

Figure 5.47 shows the overall concept to the approach. The objective is to separate the PM and OM, meaning that the OM can be simulated only once with prediction isolated to the PM.

In this way, the process for the end-user is separated into two components:

i) *One-time CFD* simulation of the surrounding context OM, without the focal design model;

ii) *Repeatable ROM* predictions on iterations of a design model PM, using the flow field wind speed data from the OM as predictor feature;



(a) Training (PM) (b) Test (OM) (c) Validation (OM+PM)

Figure 5.47: Local interference general method principal.

The two extra steps not required of the end-user are training and validation. The first involves multiple simulations on the procedural model, as before, but now for a range of uniform wind profile speeds. The second means running a simulation on the context and design models together to calculate the prediction error. Note that here in the first test, the design stand-alone and procedural stand-alone training models are the same. That is, the training model and PM are the same. Figure 5.48 shows the entire process, developed from the previous section (Figure 5.10).

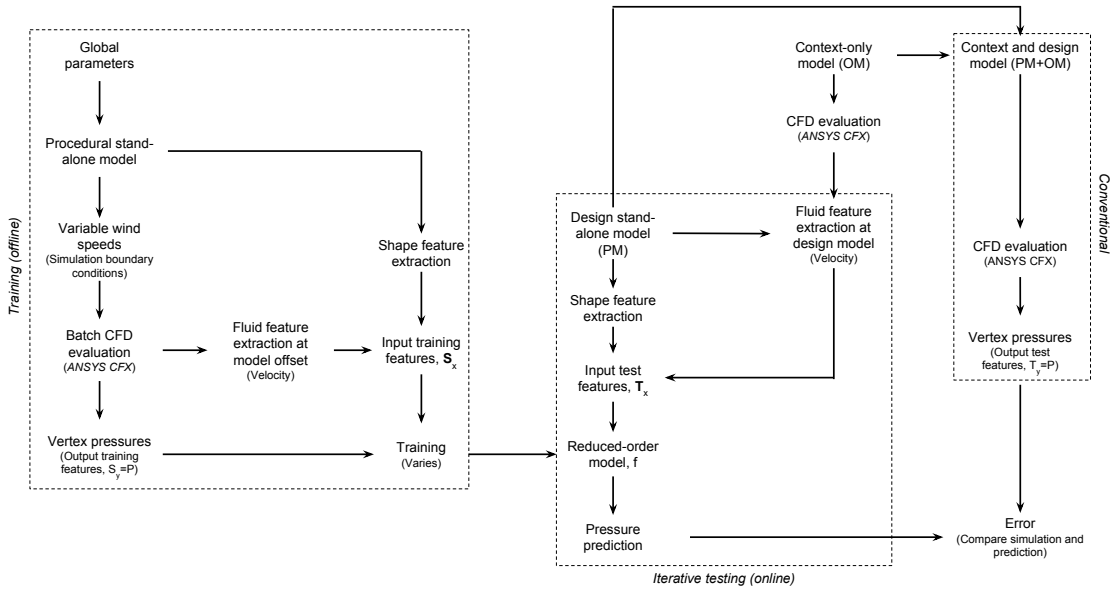


Figure 5.48: Local interference learning methodology schematic.

The *Offline* part components of the process need only be completed once, i.e. training and OM evaluation; and the *Iterative Testing* part (PM predictions) can be repeated as many times as there are different PMs, compared to the traditional approach where the design and context must be

evaluated as many times as there are different designs.

The advantage of this decomposition approach is clear: instead of having to simulate the entire scene for every change in the design model, now only the costly OM CFD simulation is required once and used for making fast, repeatable ROM predictions.

For the training models, the wind speed is taken at the projected upstream positions at a distance of $25m$ (i.e., $0.5H$). In this section, for the test model the wind speed is simply measured at the exact location of the PM vertices in the OM field. A sensitivity study is run on both these factors in the following chapter for a more complex interference case (§6.3).

5.4.2 Methodology: cuboid with single upstream cuboid

The first study uses two identical cuboids at H separation: the obstruction model (OM) directly upstream of the principal model (PM). The objective is, as before, to predict the surface pressure on the PM by using the OM's flow field velocity as input. The model configuration is shown in Figure 5.49 (left: perspective, centre: elevation, right: plan), with a separation distance of $50m$; both cuboids have dimensions W:D:H=10:10:50m. With this configuration, the downstream cuboid (PM) is primarily sheltered by the upstream cuboid (OM), leading to reduced surface pressure on the PM.

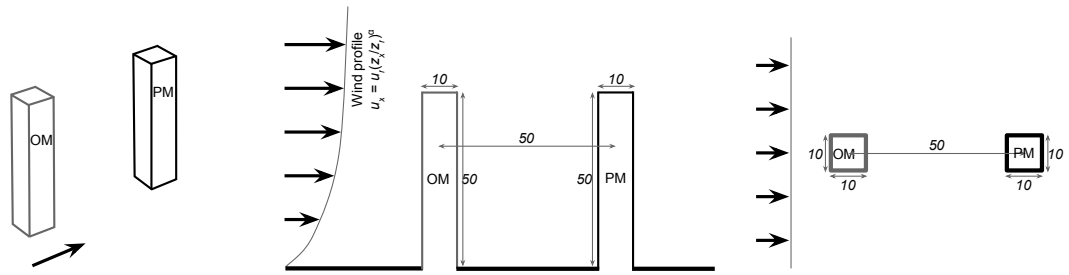


Figure 5.49: Cuboid with single upstream cuboid - model configuration.

5.4.3 Methodology: cuboid with multiple surrounding cuboids

In the second test, the OM is reconfigured with five surroundings cuboids of various sizes and locations (Figure 5.50). This is a step closer towards a realistic level of complexity, although the level of detail is still relatively basic.

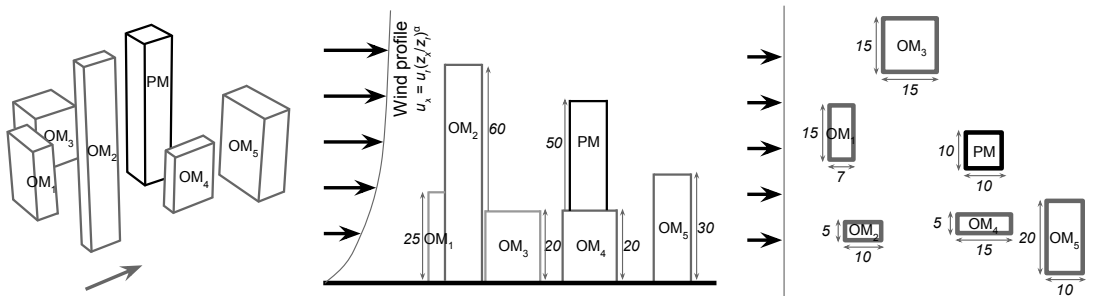
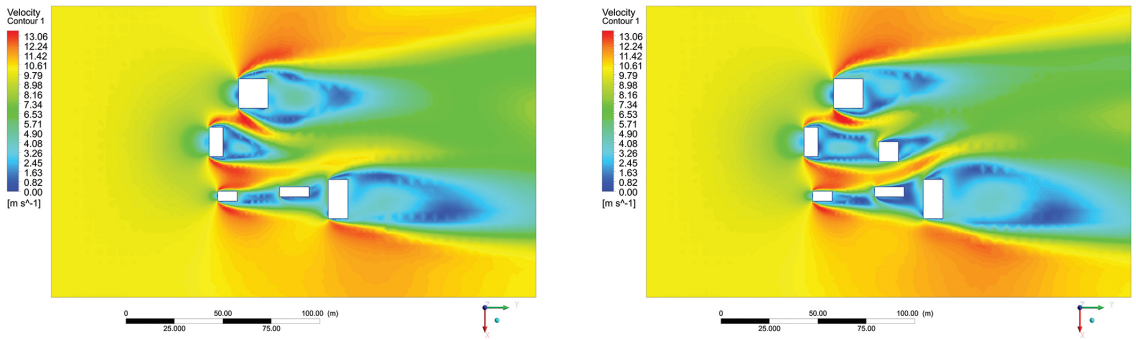


Figure 5.50: Cuboid with multiple surrounding cuboids - model configuration.

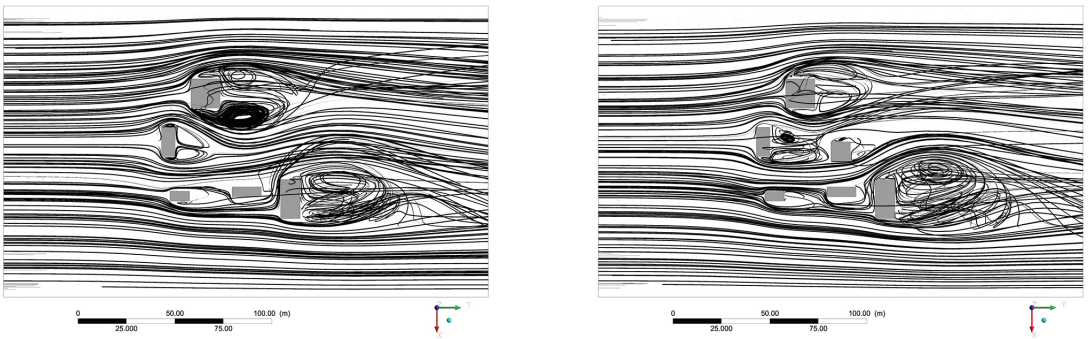
The velocity contours at the Z=10m horizontal plane are given in Figure 5.51 to compare the OM (context only) against the OM+PM (context and design for validation). The key point is that the addition of the PM does not have a uniform effect on the flow field, i.e. there are regions of both increased and decreased wind speed.



(a) OM (b) OM+PM

Figure 5.51: Velocity contours at Z=10m horizontal plane.

The difference in flow direction is shown in Figure 5.52 as streamlines, again at Z=10m. In the OM case, a large vortex or area of recirculation can be seen downstream of OM_1 which is not present in the OM+PM case.



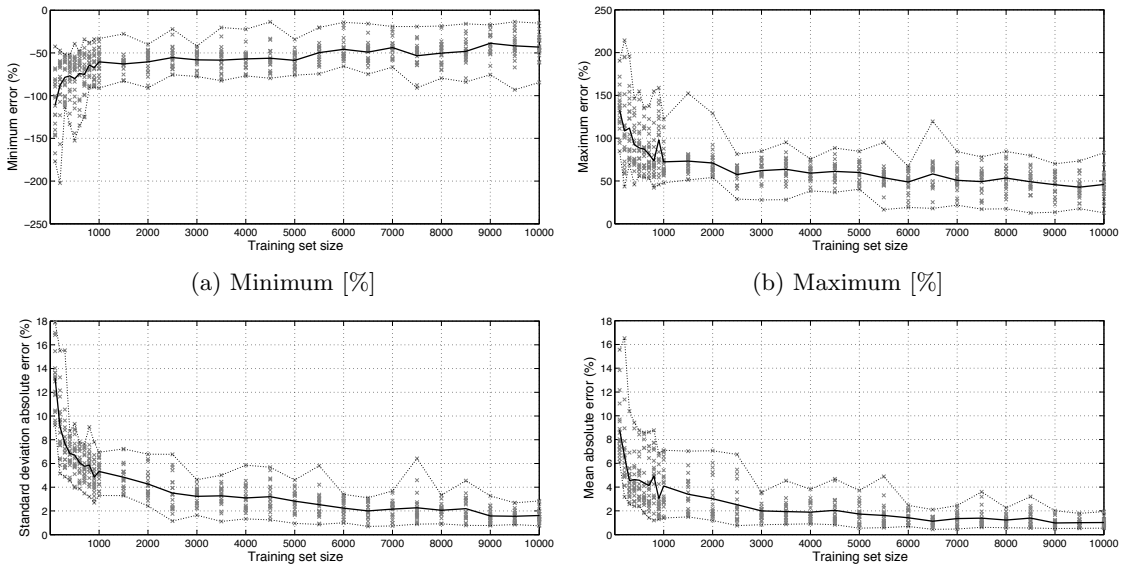
(a) OM (b) OM+PM

Figure 5.52: Streamlines at Z=10m horizontal plane.

5.4.4 Results

Results: sample-based

The sample-based training convergence errors here apply to both the following single and multi-cuboid test cases. Errors given are at the converged training set size (Figure 5.53) of n=10000.



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 5.53: Cuboid interference: sample-based training error convergence.

Sample-based training errors are: $\delta_{min.}$ = -43.144%, $\delta_{max.}$ = 45.893%, $\overline{|\delta|}$ = 1.017%, and $\sigma_{|\delta|}$ = 1.615%.

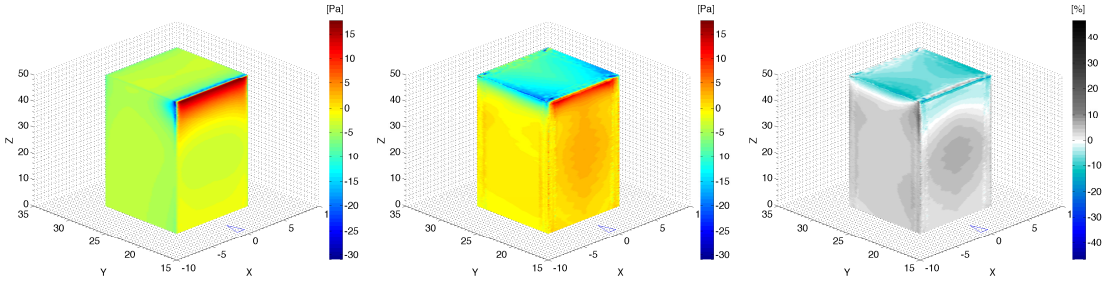
Results: model-based

There is generally good agreement in the spatial distribution of positive and negative pressure between the simulated and predicted models; although certain regions suffer from over- or under-prediction more so than for previous studies. Prediction errors and times are summarised in Table 5.15. In this study, the offline test time includes the addition of the OM simulation (further test time analysis given in §6.3).

Table 5.15: Error and time results summary - single and multi-cuboid interference: model-based.

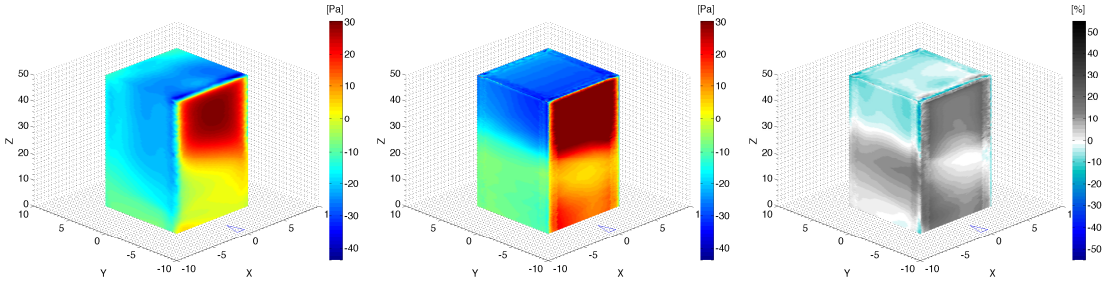
		Error, δ [%]				**Test time, t [s]**		
Case	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
Single	45093	-33.654	19.134	4.139	1.688	1255.389	0.1389	1255.528
Multi	45093	-28.175	23.306	6.744	3.985	1255.389	0.1389	1255.528

Figure 5.54 shows under-prediction on the top face for the single-cuboid OM; and Figure 5.55 shows a mixture of over-prediction (front and side faces) and under-prediction (top face) for the multi-cuboid OM.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.54: Cuboid with single upstream cuboid: test model.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 5.55: Cuboid with multiple surrounding cuboids: test model.

The predictions in Figures 5.54 and 5.55 are perhaps, qualitatively, of a lower accuracy than previous studies in this chapter. In particular, the top face of the first case and side faces of the second. However, as will be seen in the next chapter for the more complex interference case (§6.3), the addition of both a PM and an OM with greater complex appears to improve the prediction accuracy. This may be due to the increased differentiation of features of the geometry and the increased range of pressure values.

5.5 Summary

In the first study on global approximations (§5.2), two limitations to the method were identified that are addressed in the subsequent sections. Firstly, using a procedural model's global characteristics as input feature limits the flexibility of the approach. The small selection of test buildings highlights how varied tall buildings can be, so that any single procedural model attempting to represent all possible variations will inevitably be unable to do so sufficiently. For example, there are certain design features that were not included in this study, such as stepping, occlusions (self-shading or concavities), twisting, irregular floor plates, irregular vertical profile, bundling or separation. Secondly, the feature vector only predicted peak surface pressure from the global design parameters; a single value that constitutes the global performance of the model. In reality, higher resolution pressure distribution over the entire facade surface is required.

In the second (§5.3), these limitations were addressed by testing a novel feature vector methodology with intrinsic, local mesh vertex properties and individual vertex pressures. In creating a relationship between a building's geometry and its aerodynamic behaviour, the first step is to create a description of the geometry with which to relate the simulation output. By doing so the problem is redefined from a representation of the whole building model to a description of each vertex individually. A number of relatively simple cases test this methodology and establish the time and accuracy of the approach relative to the basis RANS CFD. In contrast to the top-down approach, examples are given where local geometry (vertex) descriptors are related to a local output (vertex wind pressure); showing how this localisation of the relationship increases the applicability of the method considerably.

Finally, a methodology is introduced for increasing problem complexity by the inclusion of surrounding context for urban interference (§5.4). This is again tested on two relatively simple cases (a cuboid with another single upstream cuboid; and a single with multiple surrounding cuboids) with an assessment of the prediction time and accuracy. In the following chapter, the scalability of the methodology is developed for three aspects of complexity found in realistic problems.

It can be observed that the sample-based errors, or training convergences, throughout this chapter have been relatively unremarkable. For instance, in each case, convergence is achieved with a training set size of 10000. This indicates that the training errors are not dependent on the specific problem but just on the inherent properties of the reduced-order model, i.e. the feature vector and the neural network. The sample-based errors from this chapter and the next are analysed in Chapter 7.

Chapter 6

Complex Applications

In this chapter, three complex aspects are addressed that are found in realistic problems, each of which is a development of the simpler studies from the previous chapter. The first investigates the use of time-dependent peak pressure (from LES) in unsteady turbulent flows rather than time-averaged pressure (from RANS). This is a common property of turbulent flows and is largely over-looked when using RANS steady-state simulations which produce time-averaged results. The second aspect is complex geometry common in real tall buildings. The key point of this is to show that a typically simpler procedural model can be used to generate a training set and make predictions for a more geometrically complex test set. And the third aspect is to consider realistic interference where an urban context is used alongside a representative design model.

Table 6.1: Summary of Chapter 6 studies.

	Study	Alg.	Output	Section
Application scalability				
	Time-dependent peak pressure	ANN	Vertex	6.1
	Complex geometry	ANN	Vertex	6.2
	Complex interference	ANN	Vertex	6.3

Each is a test of the scalability of the methodology described in the previous chapter by moving from tests on simple problems to more complex ones as will be found in practice.

6.1 Time-Dependent Peak Pressure

An important element of turbulent fluid flow is its change over time, not with changing boundary conditions but through periodic unsteady flows. So far only steady-state or time-averaged RANS simulations have been used. This is typically both necessary to make the computation possible, and useful to simplify a complex situation. However, as engineering usually considers worst-case scenarios in order to avoid structural failure it is necessary to predict the maximum pressure (over time) instead of the average.

For certain scenarios even if the boundary conditions are steady, then transient, or time-dependent, flows can form. This is particularly likely with bluff bodies, where sharp edges can create detached

flows and vortices downstream. When an object undergoes such vortex shedding, the pressure on the lateral sides of the object fluctuates over time.

It is therefore necessary to prove that the methodology presented here works with maximum pressure values at each vertex rather than averaged ones. This does not alter the machine learning part of the process, except in replacing $mean[P(t)]$ with $max[P(t)]$ or $min[P(t)]$ (since we are concerned with maximum positive and minimum negative pressure). In order to get $max[P(t)]$ and $min[P(t)]$, a transient simulation must be run, for which large eddy simulation (LES) can be used. This gives a pressure at every vertex at each time-step. From this the maximum and minimum pressures can be extracted.

The feature response Y is altered from the time-averaged pressure derived from the RANS simulations:

$$f : (Z, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{T}) \rightarrow mean[P(t)] \tag{6.1}$$

to either the minimum (§6.1.2) or maximum pressure (§6.1.3) derived from the LES:

$$f : (Z, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{T}) \rightarrow min[P(t)] \tag{6.2}$$

$$f : (Z, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{T}) \rightarrow max[P(t)] \tag{6.3}$$

6.1.1 Methodology: large eddy simulation (LES)

The fundamentals of LES have been described previously in Section 3.2.3. Parameters used for the LES are given in Table 6.2 (where different from the RANS setup given in Table 4.3). Measurements ran from $3s$ to $10s$ at $0.2s$ intervals, giving 36 individual recorded time-steps. The turbulence model used, WALE, is the Wall-Adapting Local Eddy-viscosity model.

Table 6.2: Additional LES parameters.

Parameter	Value / description	Units
Analysis type	Transient	-
Initial Time	3	s
Total Time Duration	10	s
Time Steps	0.2	s
Turbulence Model	LES WALE	-

The simulations required between 107 and 194 minutes for convergence. For this study, the cuboid orientation interpolation will be repeated, except using minimum and maximum pressure instead of the average. Therefore, all of the cuboid orientations were simulated again with the LES (0° to 85° at 5° intervals).

To demonstrate the pressure variance with time across all of the vertices on one model, Figure 6.1 gives the probability density function of the pressures across all the vertices on the 0° model. Each of the 36 separate lines shows a different time-step and its pressure distribution.

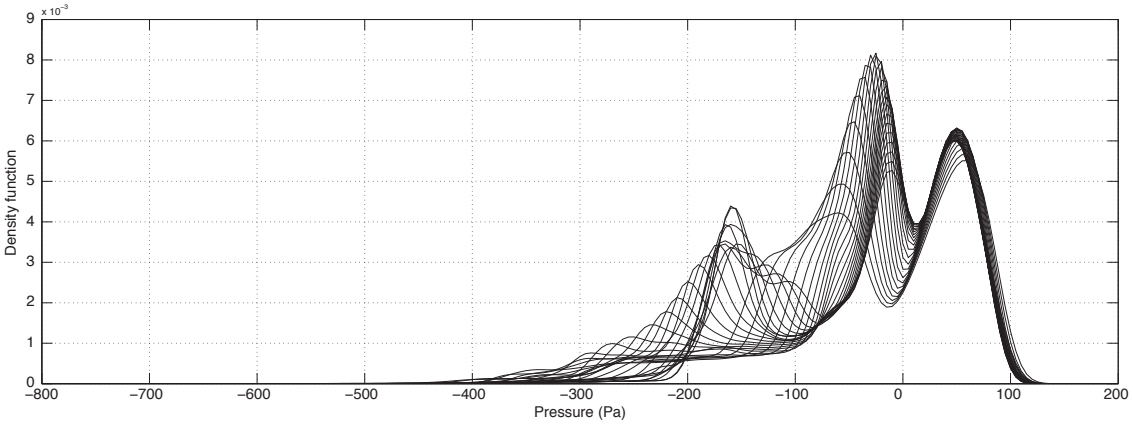
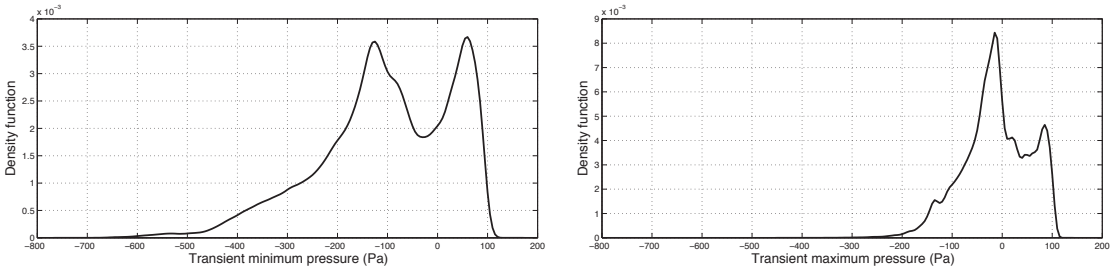


Figure 6.1: Cuboid orientation pressure probability distribution for each time-step.



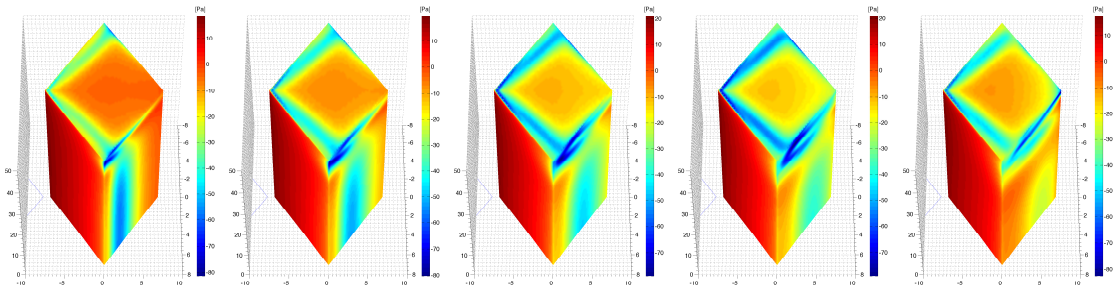
(a) (b)

Figure 6.2: Cuboid orientation pressure probability distribution: (a) $min[P(t)]$; and (b) $max[P(t)]$.

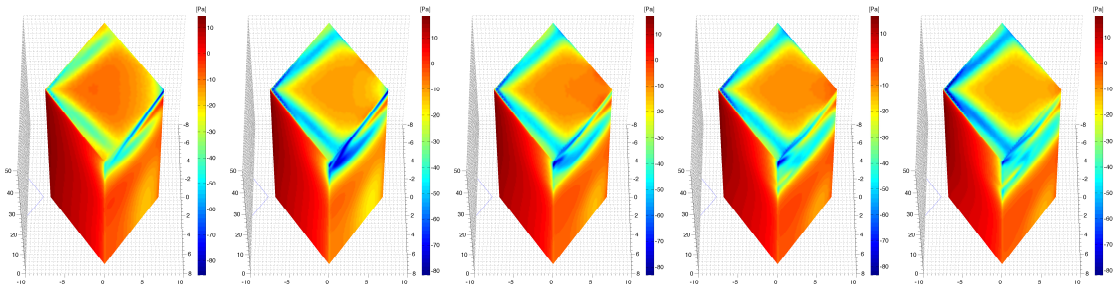
Figures 6.2a and 6.2b show the minimum and maximum pressure distributions. The minimum and maximum is calculated across all time-steps for each vertex. The code used for extracting the vertex's pressures is in Appendix B.5.

Surface pressure on the 45° oriented cuboid at a series of time-steps t throughout the transient simulation is shown in Figure 6.3. The pressure scale for each is locally defined to the minimum and maximum range for that time-step to visually emphasise the differences.

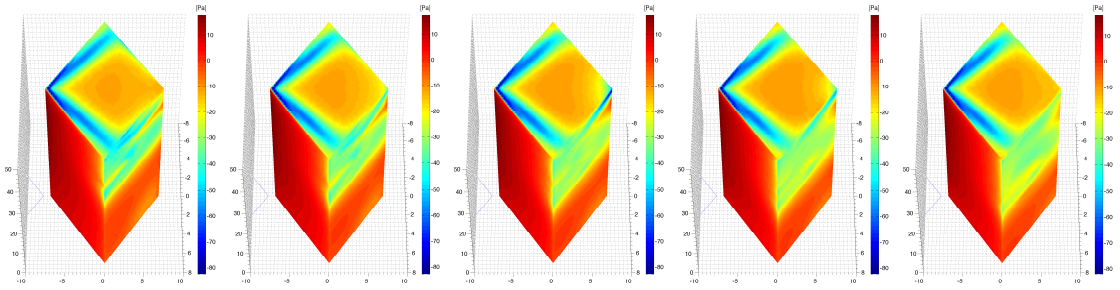
Frequency, magnitude, and stability of transient surface pressure fluctuations can change significantly between each individual case; therefore, assigning a range of time-steps from which to calculate the minimum or maximum pressure should be assessed for each case.



(a) t=4 (b) t=6 (c) t=8 (d) t=10 (e) t=12



(f) t=14 (g) t=16 (h) t=18 (i) t=20 (j) t=22



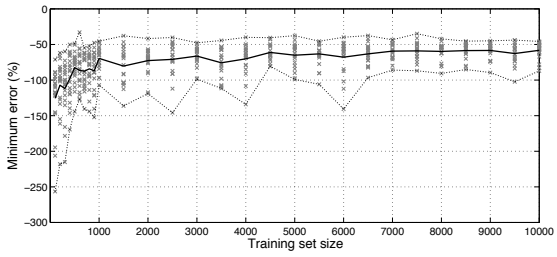
(k) t=24 (l) t=26 (m) t=28 (n) t=30 (o) t=32

Figure 6.3: Transient surface pressure on cuboid oriented at 45° for time-steps t=4,6,8,...,32.

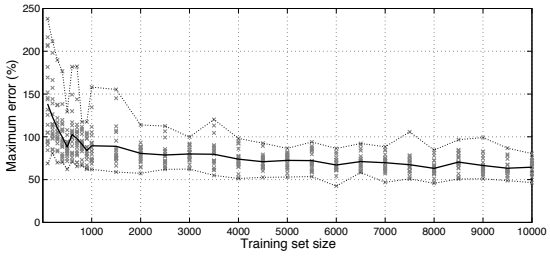
6.1.2 Minimum pressure over time

For each vertex, the minimum pressure (peak negative) across all time-steps is found, i.e. $min[P(t)]$.

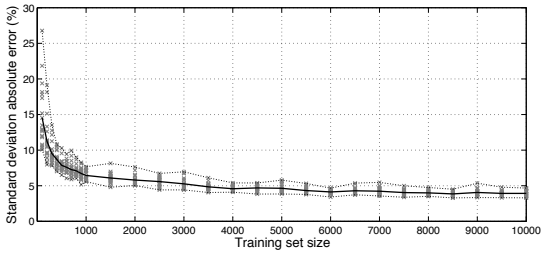
Results: sample-based



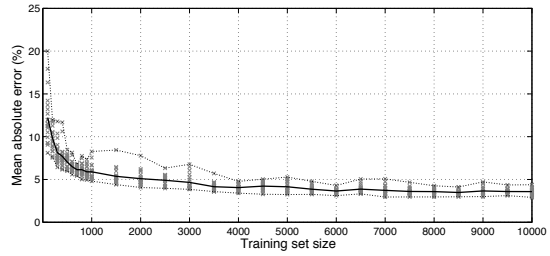
(a) Minimum error [%]



(b) Maximum error [%]



(c) Standard deviation absolute error [%]



(d) Mean absolute error [%]

Figure 6.4: Cuboid orientation $min[P(t)]$: sample-based training error convergence.

Each shows the training convergence from 20 individual runs (grey points), the error range (dotted

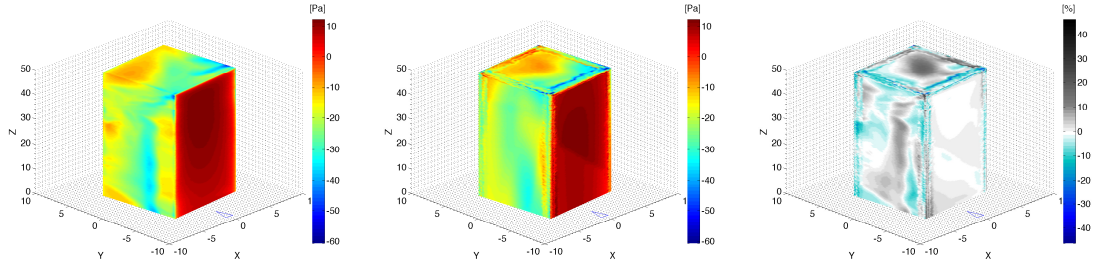
lines), and the mean (solid line). The errors given are at the converged training set size (Figure 6.4) of n=10000. The sample-based errors are: $\delta_{min.}$ = -58.284%, $\delta_{max.}$ = 64.514%, $\overline{|\delta|}$ = 3.580%, and $\sigma_{|\delta|}$ = 3.921%.

Results: model-based

Table 6.3: Cuboid orientation $min[P(t)]$: model-based ROM time vs. errors [%].

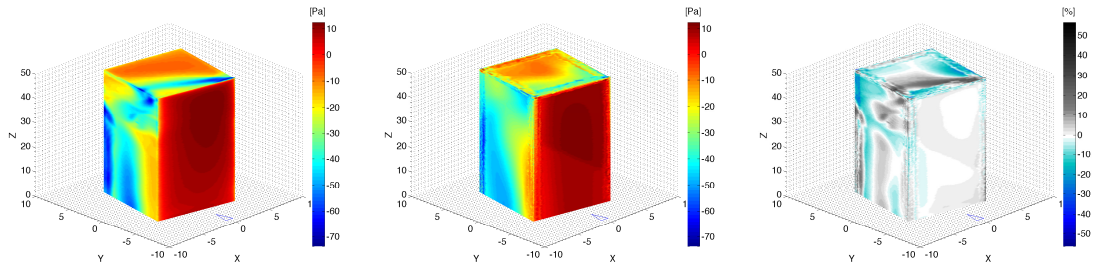
		Error, δ [%]				**Test time, t [s]**		
Orientation	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
(0)	45087	-	-	-	-	-	-	-
5	41931	-49.490	40.744	3.563	3.511	1167.359	0.1207	1167.480
10	44537	-47.511	47.796	4.156	3.966	1239.910	0.1297	1240.040
(15)	45736	-	-	-	-	-	-	-
20	46372	-41.917	50.139	4.140	4.555	1290.996	0.1342	1291.131
25	48447	-42.198	49.324	3.843	4.513	1348.764	0.1317	1348.896
(30)	49653	-	-	-	-	-	-	-
35	50336	-35.198	51.855	3.862	3.924	1401.354	0.1407	1401.495
40	51177	-33.432	45.782	3.791	3.812	1424.768	0.1493	1424.917
(45)	52517	-	-	-	-	-	-	-
50	51186	-35.918	49.404	3.729	3.759	1425.018	0.1407	1425.159
55	50314	-39.853	53.510	3.694	4.104	1400.742	0.1418	1400.884
(60)	49657	-	-	-	-	-	-	-
65	48472	-49.209	51.235	3.608	4.428	1349.460	0.1363	1349.597
70	46318	-43.272	54.341	3.986	4.545	1289.493	0.1234	1289.617
(75)	45659	-	-	-	-	-	-	-
80	44630	-50.082	53.672	4.236	4.165	1242.499	0.1219	1242.621
85	41855	-55.422	47.736	3.651	3.731	1165.243	0.1185	1165.362

The best and worst model-based predictions are shown below; the lowest mean error on 5° (Figure 6.5) and the highest on 80° (Figure 6.6).



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.5: Cuboid orientation $min[P(t)]$: min. $\overline{|\delta|}$ test model = 5°.



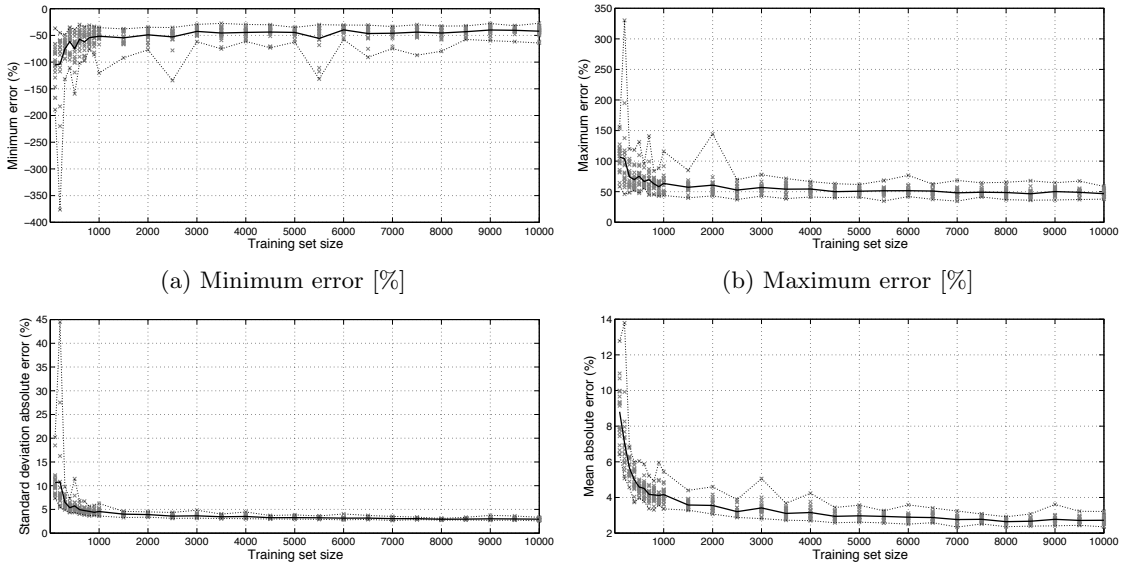
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.6: Cuboid orientation $min[P(t)]$: max. $\overline{|\delta|}$ test model = 80°.

6.1.3 Maximum pressure over time

Results: sample-based

For each vertex, the maximum positive pressure across all time-steps is found, i.e. $max[P(t)]$. The errors given are at the converged training set size (Figure 6.7) of n=10000. The sample-based errors are: $\delta_{min.}$ = -42.180%, $\delta_{max.}$ = 46.666%, $\overline{|\delta|}$ = 2.720%, and $\sigma_{|\delta|}$ = 2.947%.



(a) Minimum error [%]

(b) Maximum error [%]

(c) Standard deviation absolute error [%]

(d) Mean absolute error [%]

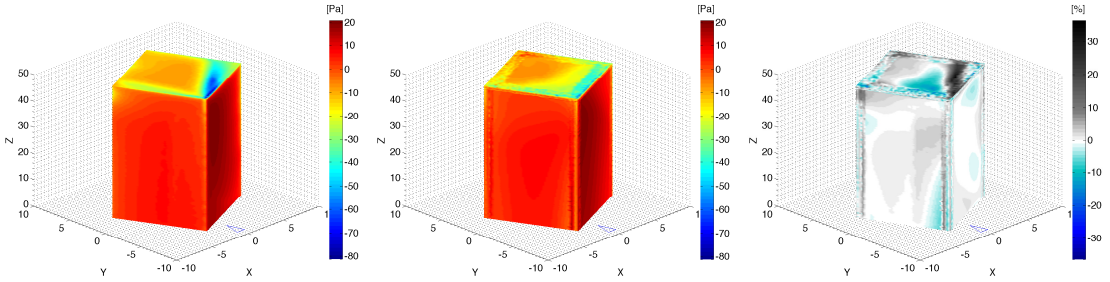
Figure 6.7: Cuboid orientation $max[P(t)]$: sample-based training error convergence.

Results: model-based

Table 6.4: Cuboid orientation $max[P(t)]$: model-based ROM time vs. errors [%].

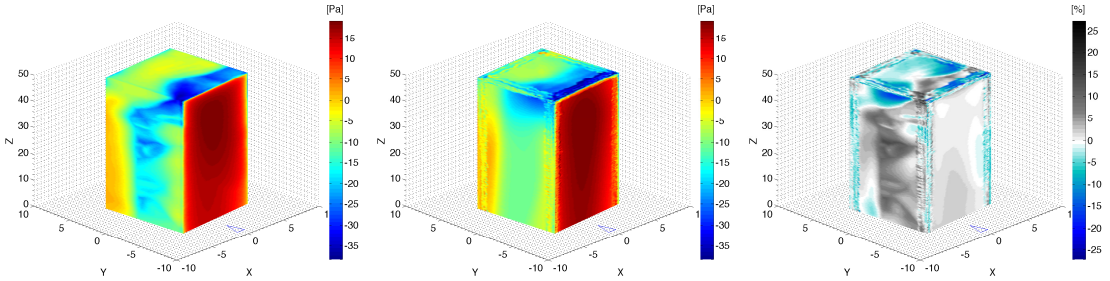
		Error, δ [%]				**Test time, t [s]**		
Orientation	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
(0)	45087	-	-	-	-	-	-	-
5	41931	-41.589	35.061	3.147	3.057	1167.359	0.1207	1167.48
10	44537	-41.894	36.982	3.130	3.219	1239.91	0.1297	1240.04
(15)	45736	-	-	-	-	-	-	-
20	46372	-31.697	55.106	2.292	2.383	1290.996	0.1342	1291.131
25	48447	-32.387	38.345	2.100	2.364	1348.764	0.1317	1348.896
(30)	49653	-	-	-	-	-	-	-
35	50336	-26.587	37.128	2.697	3.421	1401.354	0.1407	1401.495
40	51177	-33.482	40.911	2.662	3.088	1424.768	0.1493	1424.917
(45)	52517	-	-	-	-	-	-	-
50	51186	-36.593	49.412	2.531	2.948	1425.018	0.1407	1425.159
55	50314	-27.422	34.843	2.786	3.465	1400.742	0.1418	1400.884
(60)	49657	-	-	-	-	-	-	-
65	48472	-33.049	47.921	2.133	2.273	1349.46	0.1363	1349.597
70	46318	-33.061	43.425	2.357	2.372	1289.493	0.1234	1289.617
(75)	45659	-	-	-	-	-	-	-
80	44630	-53.641	34.246	3.013	3.177	1242.499	0.1219	1242.621
85	41855	-50.042	36.758	2.949	2.973	1165.243	0.1185	1165.362

The best and worst model-based predictions are shown below; the lowest mean error on 25° (Figure 6.5) and the highest on 5° (Figure 6.9).



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.8: Cuboid orientation $max[P(t)]$: min. $\overline{|\delta|}$ test model = 25°.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.9: Cuboid orientation $max[P(t)]$: max. $\overline{|\delta|}$ test model = 5°.

6.1.4 Results: ROM RANS vs. ROM LES

The reduced-order RANS model-based predictions on cuboid orientation interpolation (§5.9) have mean errors of $\delta_{min.}$ = -18.302%, $\delta_{max.}$ = 26.093%, $\overline{|\delta|}$ = 1.050%, and $\sigma_{|\delta|}$ = 1.446%. These compare to the ROM LES minimum pressure predictions with $\delta_{min.}$ = -43.625%, $\delta_{max.}$ = 49.628%, $\overline{|\delta|}$ = 3.855%, and $\sigma_{|\delta|}$ = 4.084%; and for ROM LES maximum pressure predictions with $\delta_{min.}$ = -36.787%, $\delta_{max.}$ = 40.845%, $\overline{|\delta|}$ = 2.650%, and $\sigma_{|\delta|}$ = 2.895%.

Both the ROM LES model-based prediction errors are between 2- to 3-times greater than for the ROM RANS. The ROM RANS simply takes the vertex pressures spatially across the model, whilst the ROM LES takes them spatially and temporally across the model. The difference is that the local variation of pressure across the surfaces of the LES models is greater and at a smaller scale; therefore increasing the difficulty of locally accurate predictions.

6.2 Complex Geometry

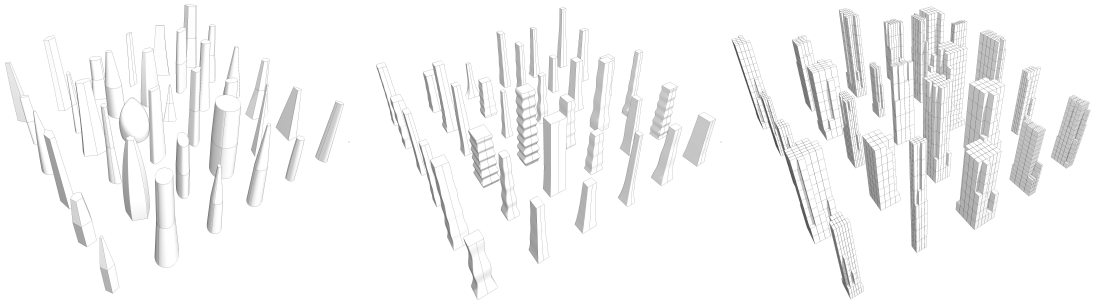
Contemporary tall building design, as discussed previously, has been freed up by computational design tools, analysis, and construction methods. The iconicity of skyscrapers is also a driving force for unique, bespoke forms; as such, increasingly complex forms are being planned and constructed. This presents an interesting challenge to wind engineers who, for the new generation

of tall buildings, typically struggle to find general behavioural rules akin to the top-down global learning approach, i.e. bespoke designs require bespoke analysis.

The following study is therefore a significant development to the previous approach described in §5.2. In a similar way, a procedural tall building model is used to generate the reduced-order model and is tested on an extended set of 10 real buildings. The differences are that the local feature vector definition is now used and the training set is also extended from 400 to 600 models with added complexity in the procedural model.

6.2.1 Methodology: procedural model

The geometry for the training set was generated using a procedural tall building model with a select number of key parameters. There are three separate topologies in the procedural model each with their own parameters, shown in Figure 6.10. The *GC* code for these models is given in Appendix B.6.



(a) Extrusion (b) Periodic (c) Blocks

Figure 6.10: Procedural training model sets.

The three procedural models can be classified as one of three topologies: *Extrusion*; *Periodic*; and *Blocks*. The topology is initially selected randomly, and then each can be used to generate instances by randomly assigning parameter values from the ranges given in Table 6.5.

Table 6.5: Procedural training model parameter ranges.

	N	W	D	H	mSF	tSF	F	o	R	A	D	f	fO	Nw	Nd	Nh
Extrusion																
Min.	3	10	10	100	0.5	0.1	0.3	2	0	-	-	-	-	-	-	-
Max.	7	60	60	200	1.1	1.1	5.1	4	180	-	-	-	-	-	-	-
Inc.	1	0.1	0.1	0.1	0.1	0.1	0.1	1	0.1	-	-	-	-	-	-	-
Periodic																
Min.	-	10	10	100	-	-	0.5	-	0	0.1	100	0.1	-180	-	-	-
Max.	-	20	20	200	-	-	20	-	180	10	1000	2	180	-	-	-
Inc.	-	0.1	0.1	0.1	-	-	0.1	-	0.1	0.1	1	0.1	1	-	-	-
Blocks																
Min.	-	10	10	100	-	-	-	-	0	-	-	-	-	4	4	5
Max.	-	60	60	200	-	-	-	-	180	-	-	-	-	7	7	15
Inc.	-	0.1	0.1	0.1	-	-	-	-	0.1	-	-	-	-	1	1	1

Notes for Table 4.3: N no. edges; W width $[m]$; D depth $[m]$; H height $[m]$; mSF mid planform scale factor; tSF top planform scale factor; F fillet radius $[m]$; o planform curvature order; R orientation $[°]$; A amplitude; D decay; f frequency; fO frequency offset; Nw no. blocks in width; Nd no. blocks in depth; Nh no. blocks in height.

With these parameter sets and ranges, the maximum number of potential instances for the three topologies respectively is: $2.56e^{16}$; $2.34e^{21}$; and $3.24e^{12}$; giving a sum of the three of $2.3448e^{21}$. Although this is the maximum number, certain combinations or regions of the parameter space lead to invalid instances which reduces the total. These are filtered out in the code simply with a *while(solid.Success==false)* statement.

A training set of 600 instances is generated, giving a sampling of $2.56e^{-17}\%$ of the maximum parameter space. This set is subsequently evaluated with *CFX* RANS as previously described (subset shown in Figure 6.11).

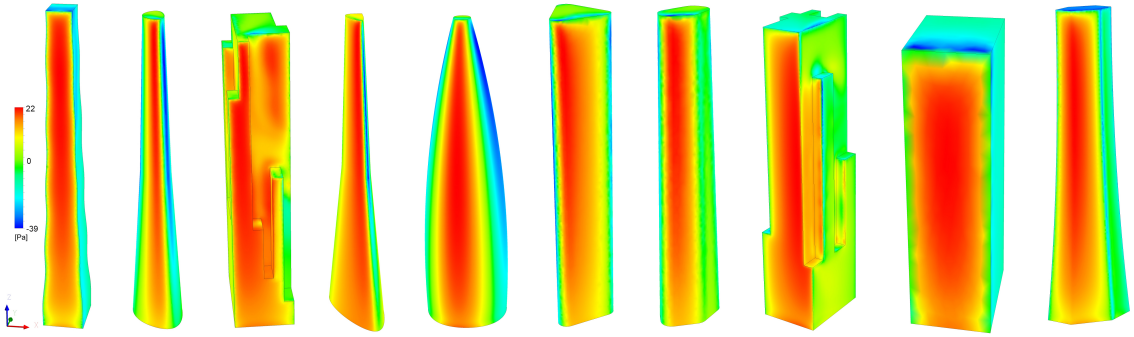


Figure 6.11: Example evaluated procedural models.

In the end, training data was collected from simulations of the 600 procedural tall building models, giving a total of D=5726831 shape features samples. As in previous cases, by randomly sampling from this full data set a far smaller training set is shown to be adequate.

6.2.2 Methodology: test set

The test set is extended from §5.2 to include three more building models (Table 6.6). The same method as before is followed: of selecting models from *Google Earth*, rebuilding them in *GC* as solids, and evaluating them with *CFX* RANS.

Table 6.6: Real building test set details.

	Name	Location	Height $[m]$	Completion date
1	Met Life Building	New York City, US	246.3	1963
2	The Shard	London, UK	306.0	2013
3	Willis Tower (Sears)	Chicago, US	442.1	1974
4	Euston Tower	London, UK	124.0	1970
5	Taipei 101	Taipei, Taiwan	508.0	2004
6	Shanghai World Financial Centre	Shanghai, China	492.0	2008
7	Bank of China Tower	Hong Kong, China	367.4	1990
8	20 Exchange Place	New York City, US	225.9	1931
9	Frankfurter Buro Center	Frankfurt, Germany	142.4	1980
10	123 Washington Street	New York City, US	192.1	2010

As before, the real building set (Figure 6.12) represents a realistically complex test of the reduced-order methodology and local shape feature definition. Crucially, the geometric complexity of the real building models is greater, and different, than that of the procedural models.

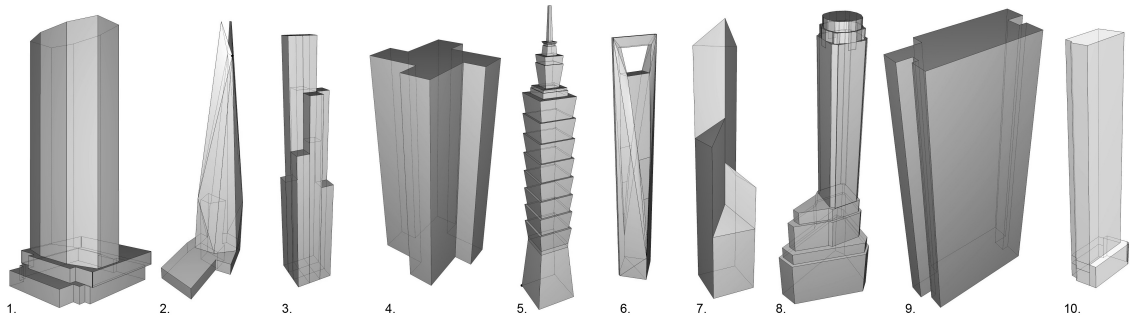


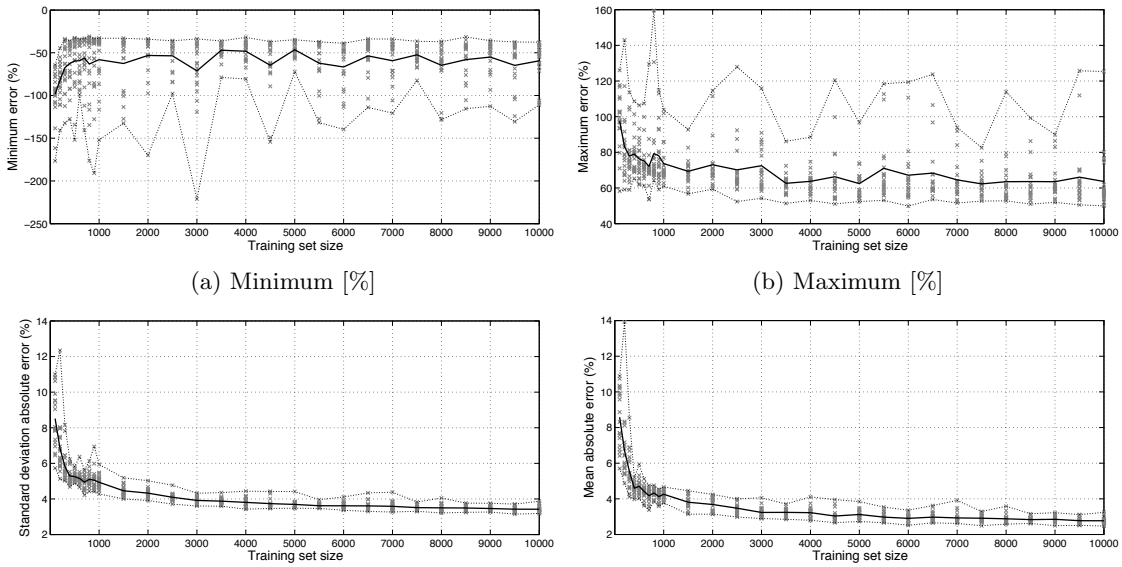
Figure 6.12: Real building test set models.

For instance, there are architectural features in the test set which are not present in the procedural model; such as base pedestals (1, 2, 8, and 10), stepping (5 and 7), and voids (6). Predictions around these design features are therefore included with the generalisations of the existing training data and subsequent reduced-order model.

6.2.3 Results

Results: sample-based

The errors given are at the converged training set size (Figure 6.13) of n=10000. The sample-based errors are: $\delta_{min.}$ = -59.365%, $\delta_{max.}$ = 63.634%, $\overline{|\delta|}$ = 2.767%, and $\sigma_{|\delta|}$ = 3.420%. The sample-based error distribution of simulated Y vs. predicted Y' pressures is given in the following chapter (Figure 7.4b).



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

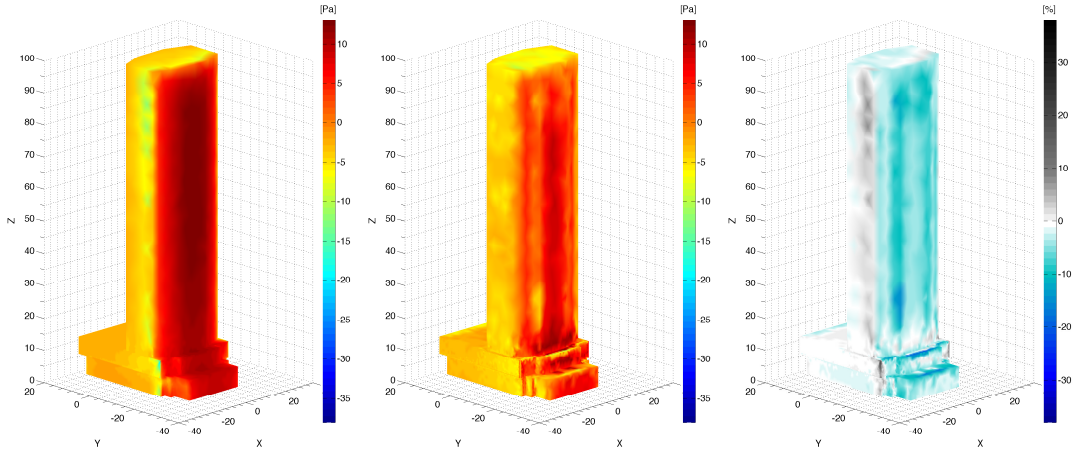
Figure 6.13: Complex geometry: sample-based training error convergence.

Results: model-based

The test set is of mixed geometric complexity which can be seen in the number of vertices on each model, m, ranging from 528 to 29091. The mean absolute errors range from 1.994% σ:2.096% (Frankfurter, Figure 6.22) to 4.440% σ:4.840% (Shanghai, Figure 6.19).

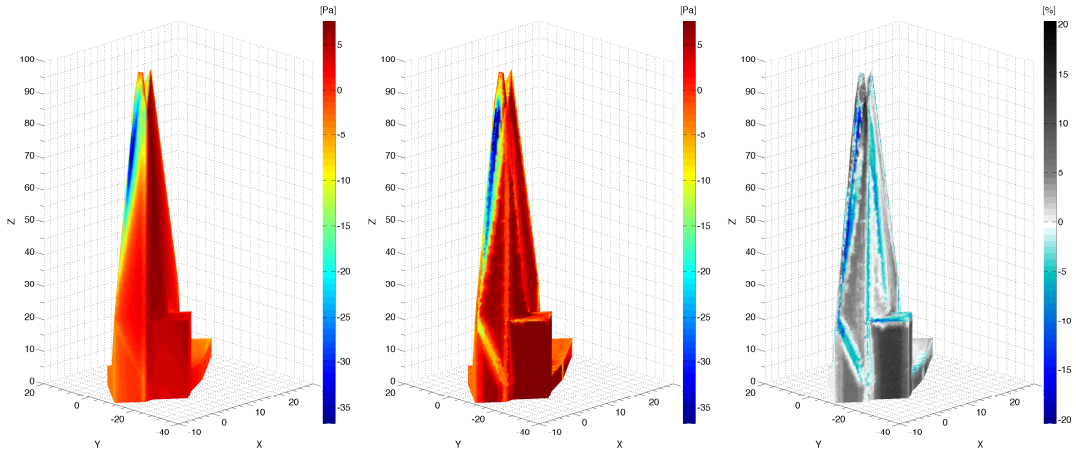
Table 6.7: Complex geometry: model-based ROM time vs. errors [%].

		Error, δ [%]				**Test time, t [s]**		
Case	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
Met Life	11439	-25.018	33.058	3.629	5.088	318.462	0.0468	318.509
Shard	20145	-21.899	13.849	2.392	2.498	560.837	0.0680	560.905
Sears	1629	-16.025	9.662	2.467	2.838	45.351	0.0230	45.374
Euston	1981	-19.770	12.314	3.669	4.059	55.151	0.0238	55.175
Taipei101	29091	-17.352	15.027	2.492	2.871	809.893	0.0898	809.983
Shanghai	10469	-23.578	25.268	4.440	4.840	291.457	0.0445	291.501
BankOfChina	528	-13.702	4.929	2.237	2.479	14.700	0.0203	14.720
Exchange	3931	-23.616	23.699	3.929	4.451	109.439	0.0286	109.468
Frankfurter	3688	-16.617	13.299	1.994	2.096	102.674	0.0280	102.702
Washington	1837	-19.099	13.380	2.766	2.225	51.142	0.0235	51.166



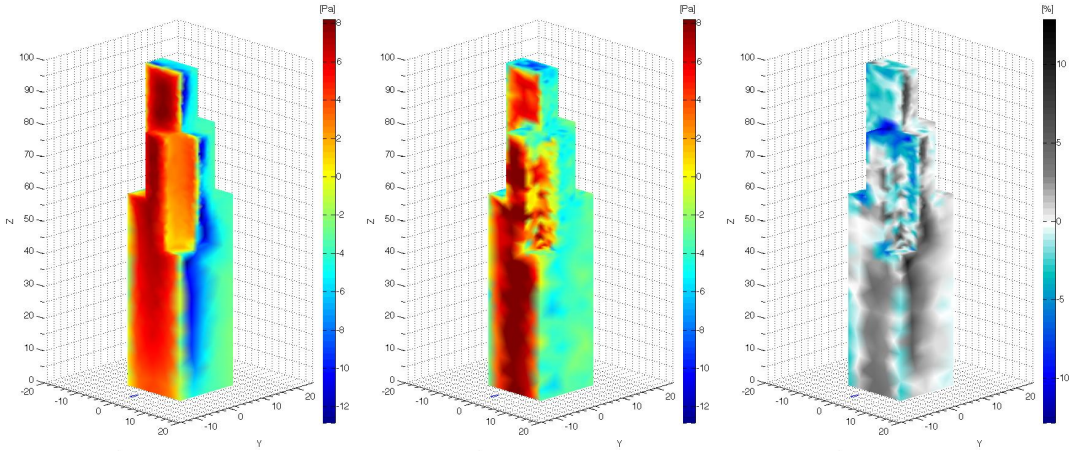
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.14: Complex geometry: test model Metlife.



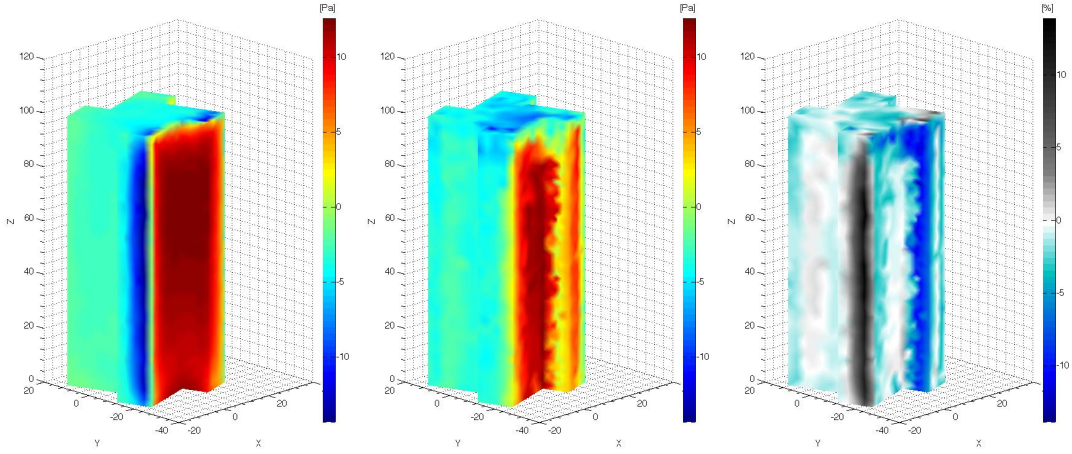
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.15: Complex geometry: test model Shard.



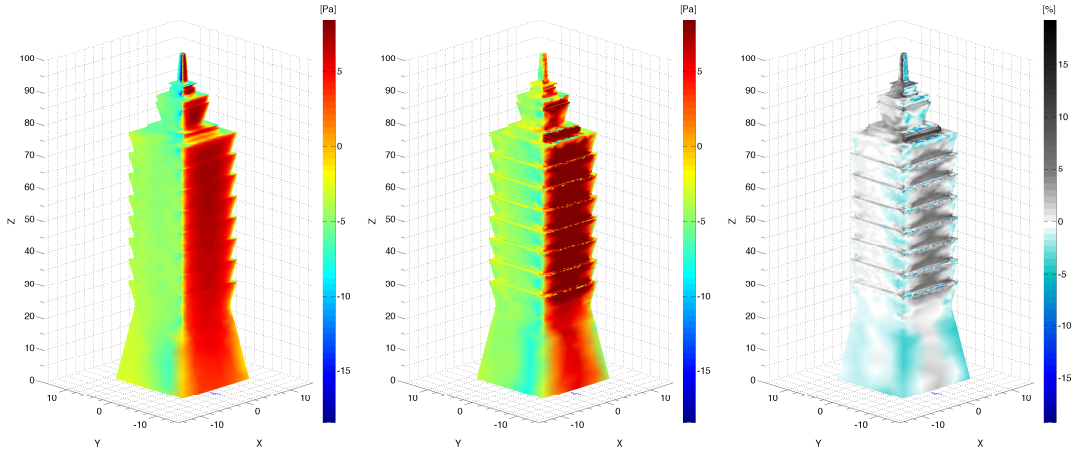
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.16: Complex geometry: test model Sears.



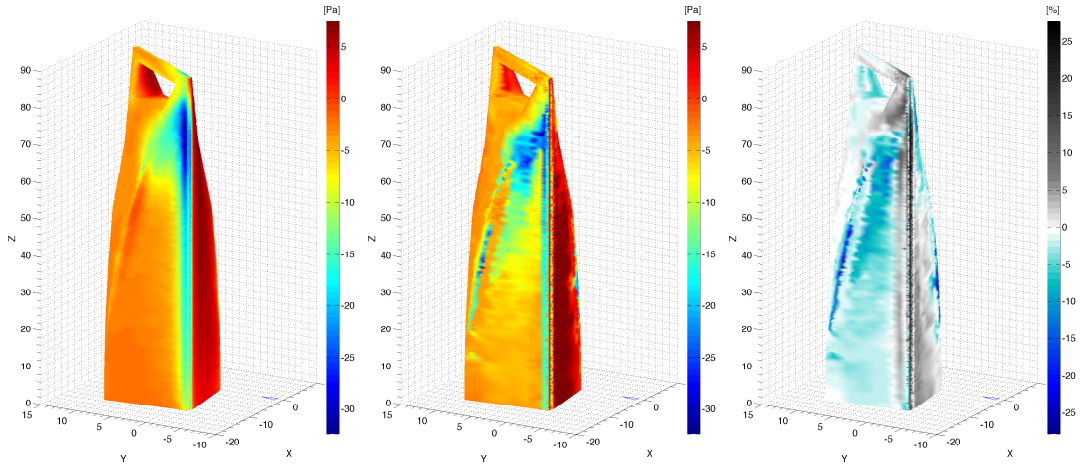
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.17: Complex geometry: test model Euston.



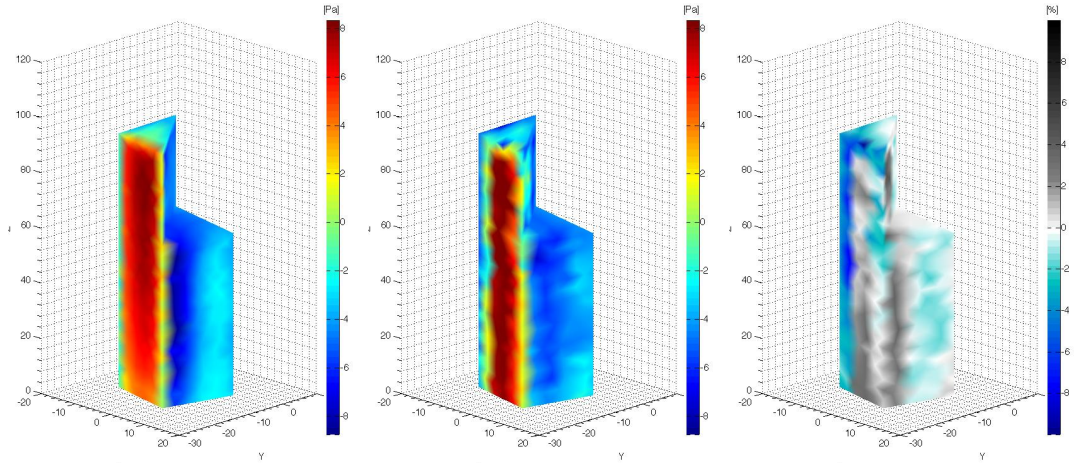
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.18: Complex geometry: test model Taipei 101.



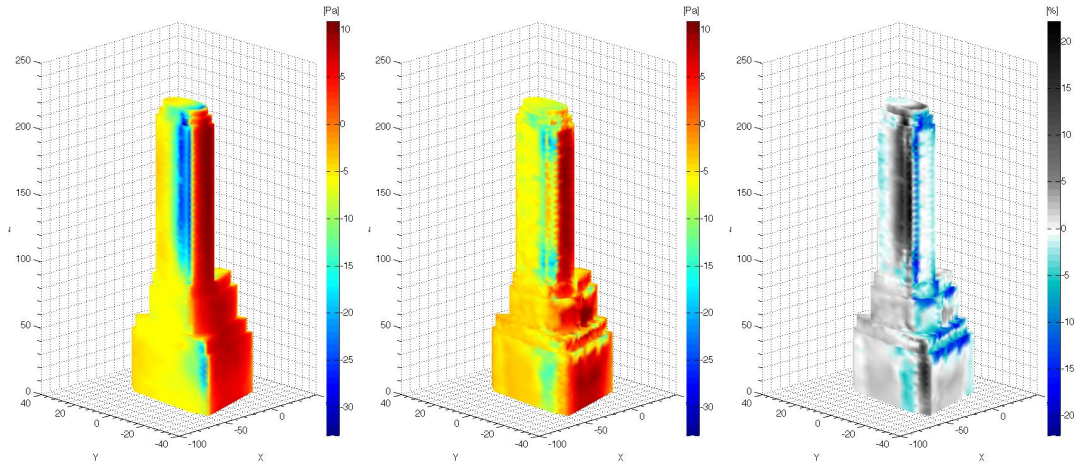
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ $[\%]$

Figure 6.19: Complex geometry: test model Shanghai.



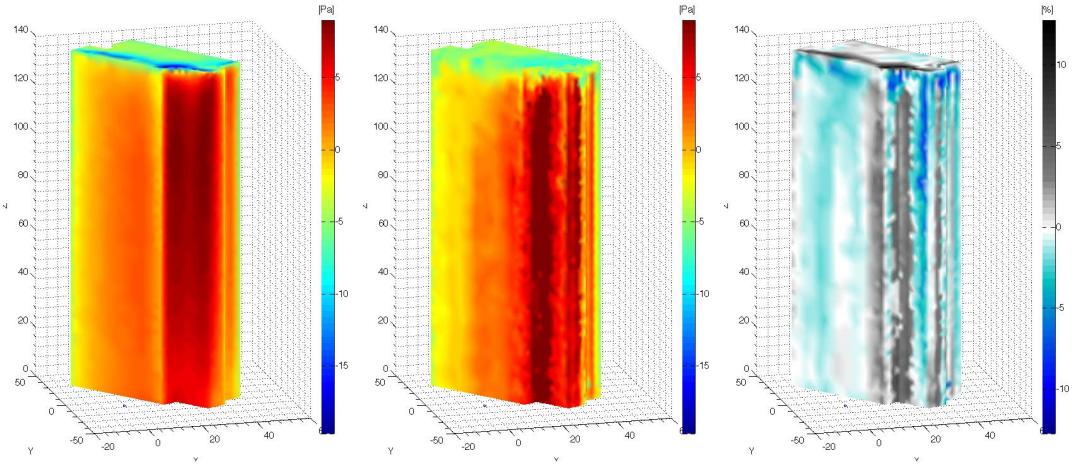
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ $[\%]$

Figure 6.20: Complex geometry: test model Bank of China.



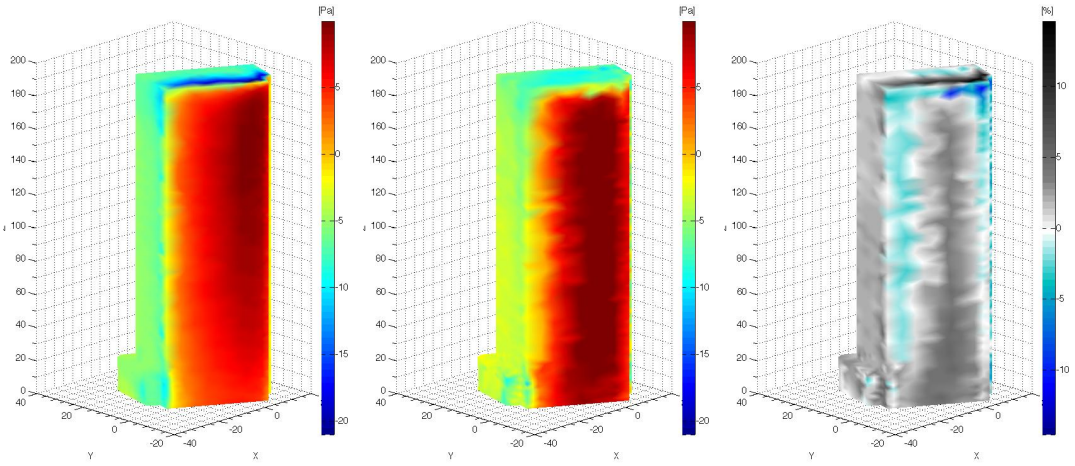
(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ $[\%]$

Figure 6.21: Complex geometry: test model Exchange.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.22: Complex geometry: test model Frankfurter.



(a) Y $[Pa]$ (b) Y' $[Pa]$ (c) δ [%]

Figure 6.23: Complex geometry: test model Washington.

A noticeable trend in these 10 test cases is of under-prediction of negative pressure values, especially in localised regions of very low pressure. To be clear, an under-prediction of a negative value means the prediction is too high, e.g. simulated value is -10 but predicted value is -5. This is a common trend across most of the cases so far and is discussed in the following chapter (Figure 7.4).

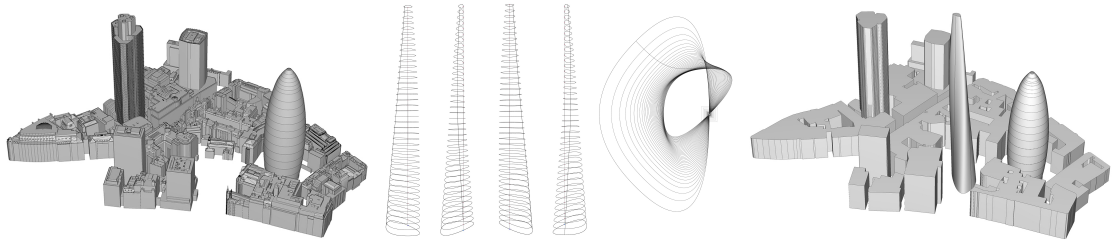
The success of the predictions can, given the early-stage design application, particularly be observed in the visual similarity between the simulation (Y, left) and the prediction (Y', centre). In this qualitative assessment of the predictions, it is both the magnitude and spatial distribution of pressure that is important, both of which are accurate enough to give a faithful representation of the simulated result.

6.3 Complex Interference

The final study intends to replicate a scenario that would be found in practice; of an early stage tall building design model in an urban context. The problem is a direct development of the methodology established in §5.4.

6.3.1 Methodology

A realistic OM of the dense City district in London is used (Figure 6.24a), along with a realistic design PM (Figure 6.24b). These are put together for the full validation model, OM+PM (Figure 6.24c). Note that the geometry in Figure 6.24a also shows the level of detail typically found from source, and Figure 6.24c shows the same geometry after simplification and meshing. The geometry intends to replicate a scenario that would be found in practice. The design model (Figure 6.24b) is arbitrary, but is based on prior models generated at competition, massing, or form-finding project stages within practice. The design model has a height of $310m$ (Z-axis), a cross-wind (X-axis) width of $55.4m$, and an along-wind (Y-axis) depth of $41.8m$; the aspect ratio (width:height) is therefore roughly 1:6. In comparison, the upstream Swiss Re is $180m$ and the downstream Tower 42 is $183m$.



(a) OM (b) PM (c) OM+PM

Figure 6.24: Components of the principal model (PM) and obstruction model (OM).

In the test case (the OM simulation), the wind speed is applied at an upstream inlet with a reference speed (v_r) of $10m \cdot s^{-1}$ at a reference height (z_r) of $10m$. The most commonly used distribution of wind speed with height is the 'power-law' expression:

$$v_x = v_r \cdot (z_x/z_r)^{\alpha} \tag{6.4}$$

The exponent α is an empirically derived coefficient that is dependent on the stability of the atmosphere. For neutral stability conditions it is approximately 0.143, and is appropriate for open-surroundings such as open water or landscape (Hsu et al., 1994). In the training models a constant vertical wind profile is used, albeit with varying speeds, so as to generate a range of upstream wind speeds across the simulated training set for every vertex, i.e. $v_x = v_r$.

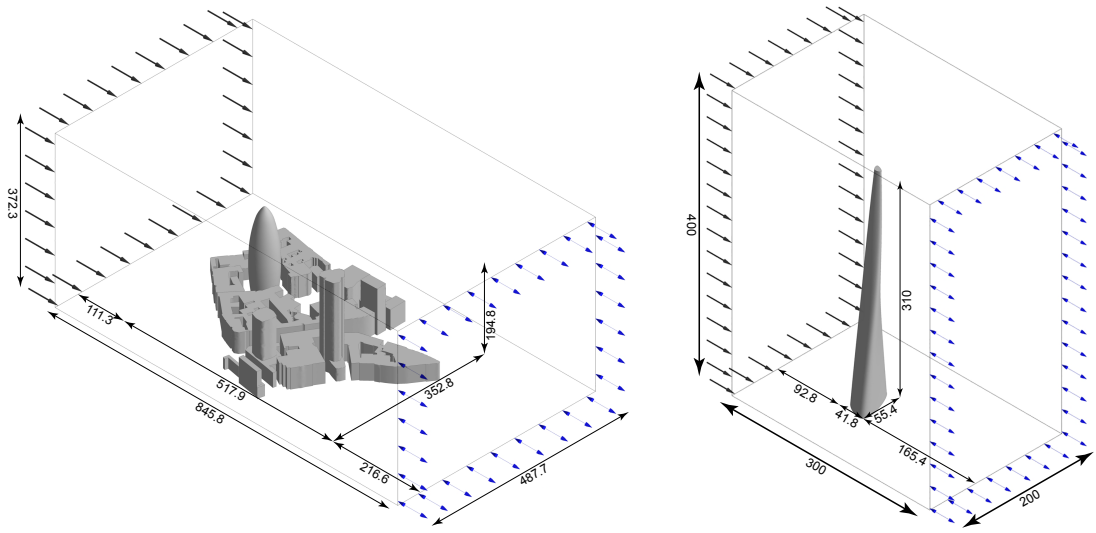


Figure 6.25: Simulation domain sizes: (left) OM; (right) PM.

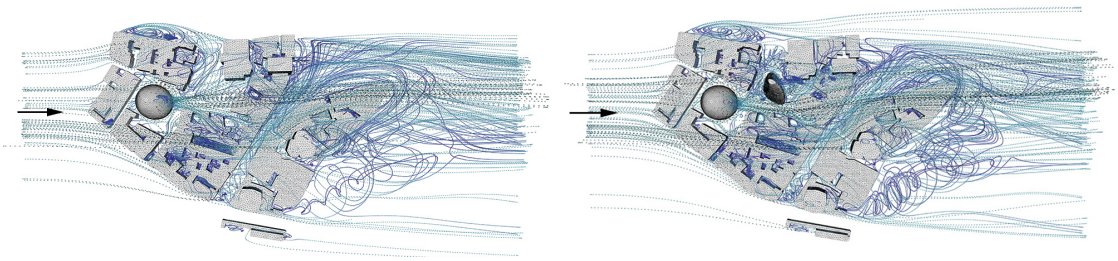


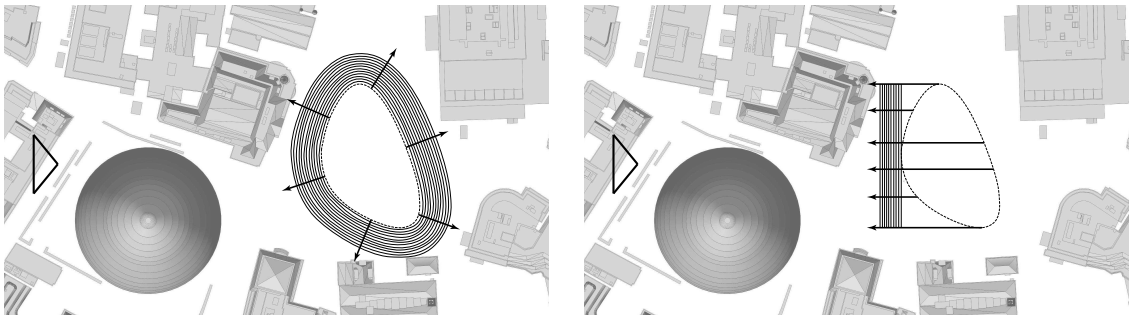
Figure 6.26: CFD flow field of (a) OM for testing and (b) OM+PM for validation.

For a training set, $\mathbf{S}$, consisting of vertex feature vectors and simulated pressure extracted from the CFD, the ANN approximates the function $f^{ANN} : \mathbf{X} \rightarrow P$ where $\mathbf{X}$ is the vertex feature vector and P is the vertex pressure. $\mathbf{X}$ is defined as:

$$\mathbf{X}\{v, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{u}\} \tag{6.5}$$

where $\mathbf{n}_{x,y,z}$ are the vertex normal components; $\mathbf{n}\sigma^{1-5}_{x,y,z}$ are the vertex-ring (one through five) neighbourhood curvature (non-absolute) standard deviation components; and $\mathbf{u}_{x,y,z}$ are the normalised relative vertex position within the model limits. For training, v_S is simply the inlet wind speed of the training simulation which is constant with height, i.e. no profile. For testing however, this is replaced with v_T the wind speed at the vertex's transformed position in the OM fluid field.

The transformation of the vertex is either a normal offset or a projection upstream from the original location (Figure 6.27). This results in either $v_{T.offset}$ or $v_{T.upstream}$, both of which are tested for different distances d in the following section. For both transformations, d is increased from 0 at increments of $1m$ until an obstruction is encountered; $12m$ for the normal offset, and $9m$ for the upstream projection.



(a) Offset (b) Upstream

Figure 6.27: Test feature wind speed location from OM.

From the training feature set, the reduced-order model is generated by a back-propagation artificial neural network (ANN), with a hyperbolic tangent sigmoid transfer function (Vogl et al., 1988):

$$tansig(x) = 2/(1 + exp(-2 \cdot x)) - 1 \tag{6.6}$$

The ANN structure X:H:Y is 22:20:1, i.e. 22 input neurons, 20 hidden layer neurons, and 1 output. The sensitivity analysis on the number of neurons in the hidden layer, and the number of layers, is not included here; although 20 in a single layer has been seen to be sufficient. There is no general rule for guidance to define either, necessitating sensitivity analysis for each problem.

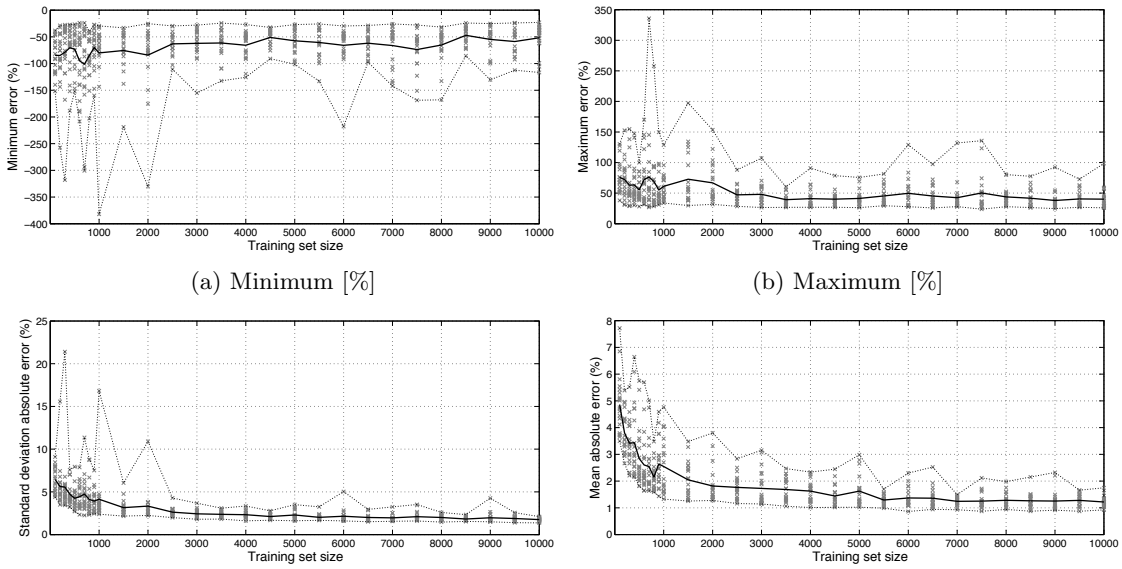
6.3.2 Results

Results: sample-based

Sample-based errors given are at the converged training set size (Figure 6.28) of n=10000. The test set size m, being the full data set D minus the training set n, is therefore $210000 - 10000 = 200000$. These are randomly selected for each training run, which is repeated 20 times. The individual runs are shown as grey crosses, with the black lines showing the limits and the blue line the mean over the 20 re-runs. The training set size is increased incrementally, by an increment of 100 from 100 to 1000, and an increment of 500 from 1000 to 10000. The converged sample-based errors are: $\delta_{min.}$ = -51.906%, $\delta_{max.}$ = 40.115%, $\overline{|\delta|}$ = 1.217%, and $\sigma_{|\delta|}$ = 1.756%.

Results: training set wind speed (v_S)

Varying the increments of inlet wind speed in the training set simulations has an impact on the time required for initially generating the ROM. The range of wind speeds is kept constant, between 1 and $15m \cdot s^{-1}$, and the increments varied between 1, 2, 7, and $14m \cdot s^{-1}$. The difference in error between an increment of 1 and $2m \cdot s^{-1}$ is minimal, yet the time saving is substantial with nearly half the number of simulations required. In fact, even with an increment of $7m \cdot s^{-1}$ the difference



(a) Minimum [%]

(b) Maximum [%]

(c) Standard deviation absolute [%]

(d) Mean absolute [%]

Figure 6.28: Complex interference: sample-based training error convergence.

in error is still minimal, but with a fifth of the time required for generating training data (Table 6.8).

Table 6.8: Error results [%] for v_S sensitivity analysis: model-based.

Inc.	v_S $[m \cdot s^{-1}]$	**No. Models**	D	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$
1	{1,2,3,...,15}	15	210000	-57.435	30.801	5.082	7.793
2	{1,3,5,...,15}	8	112000	-66.017	29.099	5.055	7.699
7	{1,8,15}	3	42000	-55.179	28.301	5.673	8.122
14	{1,15}	2	28000	-65.854	25.884	9.995	10.162

Figure 6.29 shows the probability density distribution of the wind speeds in the test data set from the OM with an offset of $0m$, i.e. no transformation. The probability density distribution uses a smoothing kernel with a normal distribution and a width of $0.02m \cdot s^{-1}$.

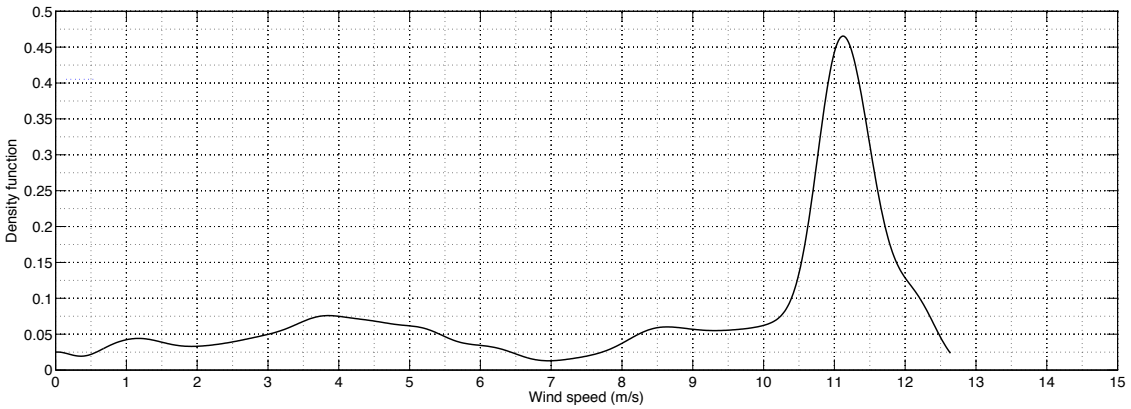


Figure 6.29: Test set wind speed probability distribution at PM.

A peak at $11.2m \cdot s^{-1}$ is clear due to the inlet wind profile reference speed of $10m \cdot s^{-1}$ at a reference height of $10m$. Further work should establish a methodology for robustly sampling wind speeds:

that is, a training set that fits the above distribution would be optimal for this case, but not for another case.

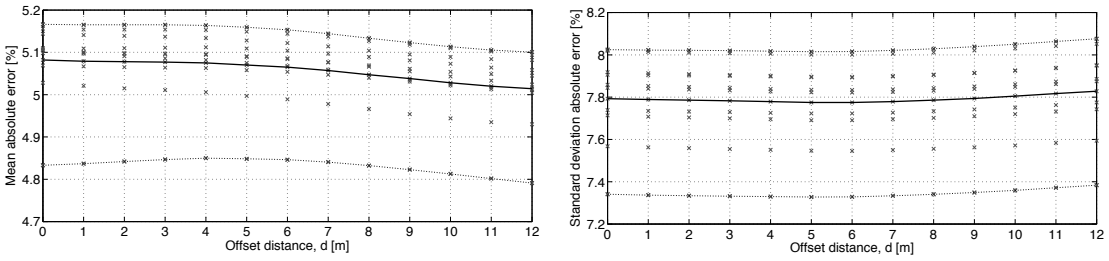
Results: test set wind speed location ($v_{T.offset}$ vs. $v_{T.upstream}$)

Two transformation methods, normal offset and upstream projection, are tested for varying distances d (Figure 6.27). The geometric transformation is applied to the PM mesh, positioned in the OM field; from which the wind speeds for the test feature vector, $v_{T.offset}$ or $v_{T.upstream}$, are calculated.

Table 6.9: Error results [%] for test location sensitivity analysis: model-based.

$v_{T.offset}$					$v_{T.upstream}$				
d [m]	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	d [m]	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$
0	-57.435	30.801	5.082	7.793	0	-57.711	30.793	5.240	7.934
1	-57.404	30.807	5.079	7.789	1	-57.796	30.793	5.232	7.942
2	-57.386	30.811	5.078	7.786	2	-57.744	30.794	5.221	7.946
3	-57.363	30.814	5.077	7.783	3	-57.688	30.794	5.209	7.952
4	-57.342	30.817	5.075	7.779	4	-57.653	30.795	5.203	7.965
5	-57.323	30.819	5.070	7.775	5	-57.669	30.795	5.201	7.981
6	-57.304	30.819	5.065	7.775	6	-57.686	30.795	5.201	7.997
7	-57.286	30.818	5.057	7.779	7	-57.705	30.796	5.204	8.012
8	-57.303	30.813	5.047	7.786	8	-57.747	30.796	5.209	8.027
9	-57.335	30.806	5.038	7.794	9	-57.828	30.796	5.216	8.045
10	-57.386	30.794	5.028	7.805					
11	-57.427	30.773	5.020	7.817					
12	-57.425	30.772	5.014	7.828					

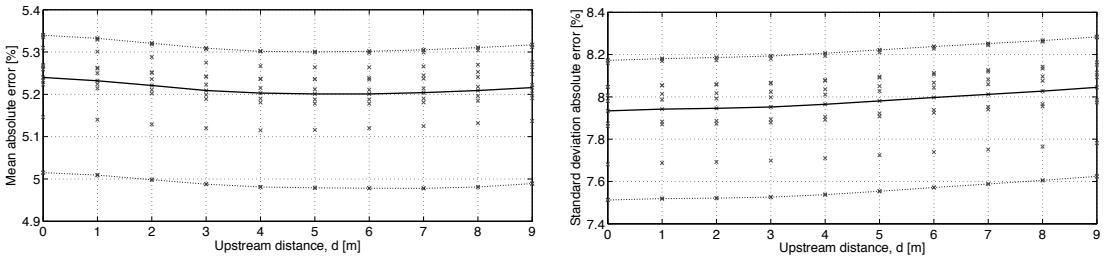
In fact, the prediction errors in Table 6.9 for both transformation methods with a varying d, suggest they are relatively invariant to the test feature wind speed location. For the offset there is a standard deviation over the range of absolute mean errors of only 0.024%, and only 0.014% for the upstream. Figures 6.30 and 6.31 plot the mean absolute (left) and standard deviation absolute (right) errors against distance for both the offset and upstream transformations. The individual runs are shown as grey crosses, with the black lines showing the limits and the blue line the mean over the 10 re-runs.



(a) Mean absolute error, $\overline{\lvert\delta\rvert}$ (b) Standard deviation error, $\sigma_{\lvert\delta\rvert}$

Figure 6.30: Test wind speed location: $v_{T.offset}$ distance vs. error.

With the normal offset transformation, the mean absolute error decreases as the offset distance increases so the minimum is at the greatest distance, d=12m. For the upstream projection transformation, the mean absolute error decreases with distance until d=5 or 6m, at which point it



(a) Mean absolute error, $\overline{|\delta|}$ (b) Standard deviation error, $\sigma_{|\delta|}$

Figure 6.31: Test wind speed location: $v_{T.upstream}$ distance vs. error.

starts to increase again.

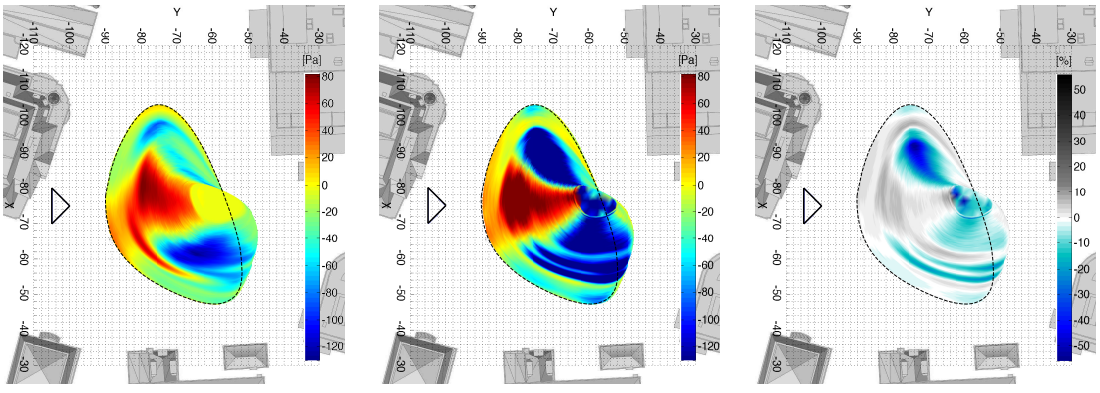
As mentioned, the variation of error with distance or transformation method is small so the optimal location at which to measure the wind speed for the test feature is still unclear. However, it is likely to be esoteric for each problem or OM and should be studied further.

Results: model-based

Table 6.10: Complex interference: model-based ROM time vs. errors [%].

		Error, δ [%]				**Test time, t [s]**		
Case	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X + Y'$
London City	14000	-57.435	30.801	5.082	7.793	389.760	0.0481	389.808

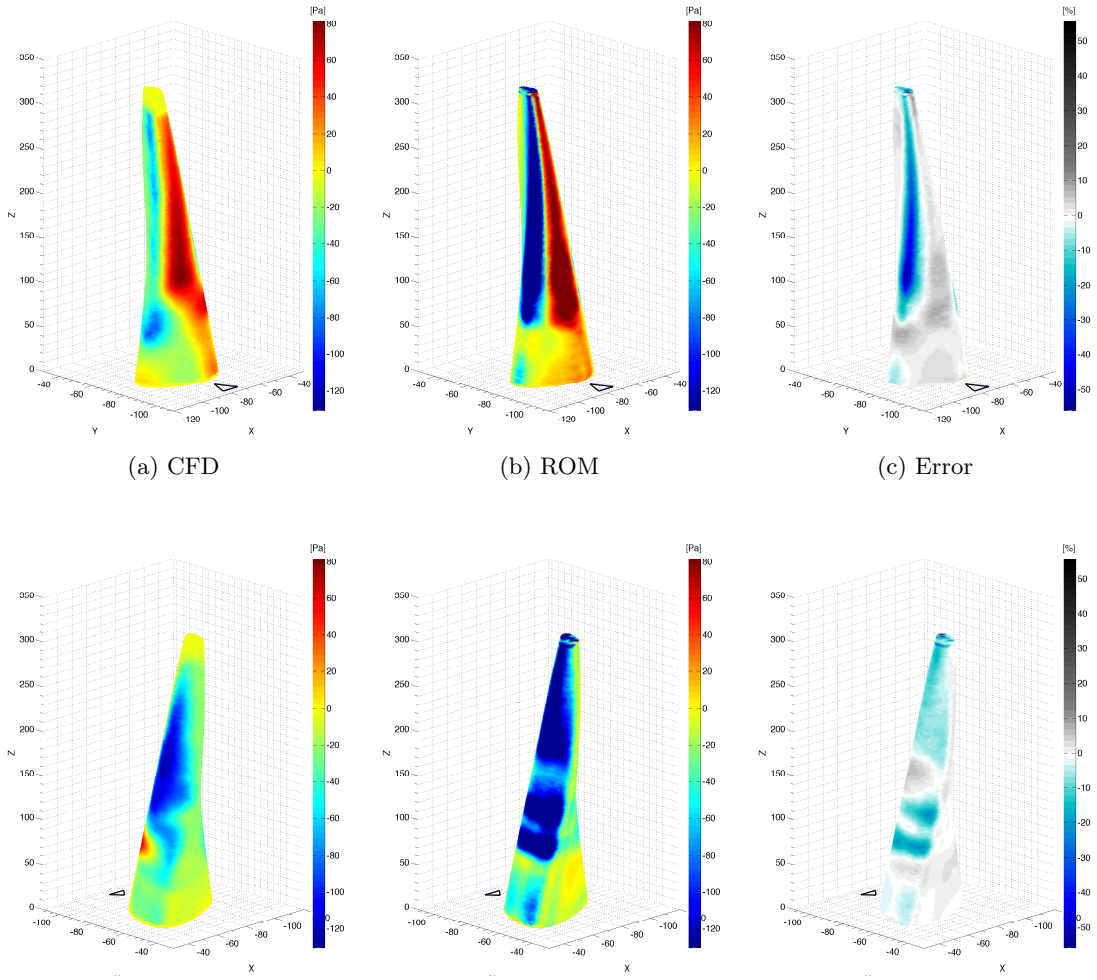
The following model-based results use n=10000, a training wind speed increment of $1m \cdot s^{-1}$, and no transformation on the test feature wind speed location. The model-based errors are: $\delta_{min.}$ = -57.435%, $\delta_{max.}$ = 30.801%, $\overline{|\delta|}$ = 5.082%, and $\sigma_{|\delta|}$ = 7.793%. Figures 6.32 and 6.33 visualise the surface pressures and errors on the PM.



(a) CFD (b) ROM (c) Error

Figure 6.32: Model-based test: plan view.

In Figure 6.33, the top three figures (a-c) are from an upstream perspective; the bottom three (d-f) from downstream. Within each triplet: the left figure is the CFD simulation (Y [Pa]), centre the ANN prediction (Y' [Pa]), and right the difference (δ [%]) between predicted and simulated



(a) CFD (b) ROM (c) Error

(d) CFD (e) ROM (f) Error

Figure 6.33: Model-based test: (a-c) upstream; (d-f) downstream side.

pressures. The inlet wind direction is indicated by an arrow on each figure.

There is generally good agreement in the spatial distribution of positive and negative pressure between the simulated and predicted models. Although the localised over-prediction (i.e. $|Y'| > Y$) of positive and negative pressure values, relative to the simulated values, is apparent in an exaggeration of the visual results.

The probability distribution, or density function, of non-absolute errors [%] for 10 re-runs (grey) and their mean (black) are shown in Figure 6.34. A training set of 10000, an increment of $1m \cdot s^{-1}$ in the training set wind speeds, and no test feature wind speed location transformation are applied.

A smoothing kernel with a normal distribution and a width of 0.01% is used. The errors fit a normal distribution, with a peak probability of about 18% that the error will be 1.1%. Towards the minimum and maximum of the error range (-10 and 10%), the probability is only between 1.5 and 0% respectively. The normal distribution of errors here is consistent with the majority

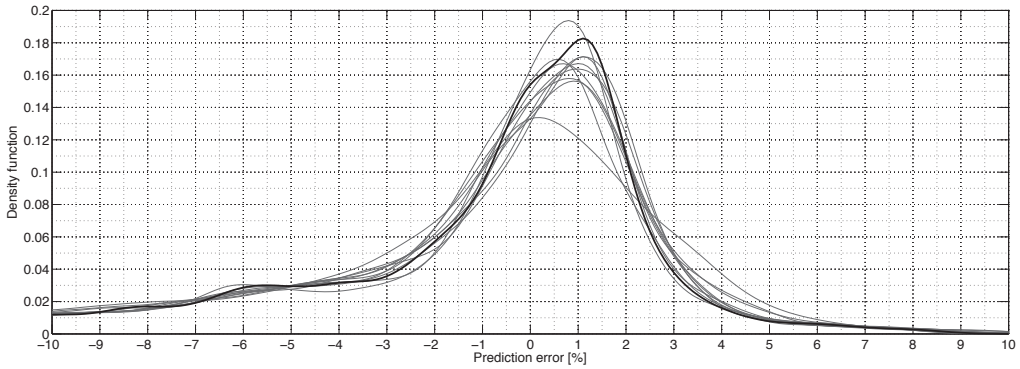


Figure 6.34: Probability distribution of prediction errors.

of the previous studies, suggesting that the prediction results and errors are generally, at least in their form if not their magnitude, are independent of the test case, but instead dependent on the features and training method.

6.4 Summary

In this chapter three components of realistic cases have been studied: i) transient simulation basis for peak pressure prediction; ii) complex geometries from a procedural model; and iii) complex urban interference. Each is an independent test of the scalability of the methodology, extending the cases in the previous chapter.

For the first (§6.1), a large eddy simulation (LES) was used to model the time-dependent flow around cuboids of different orientations. For each vertex of each model, the minimum and maximum pressure over the simulated range of time-steps were calculated. This differs from the previous chapter which use RANS, whereby the pressures are time-averaged mean values. It was found that prediction accuracy was lower than for the RANS basis, although still acceptable, because of the greater local variability in pressure distribution spatially and across time.

In the second study (§6.2), a procedural tall building model was used to generate a training set of 600 RANS evaluated models, from which the reduced-order model was generated. A test set of 10 real buildings was then used for assessment of prediction accuracy and speed. Good prediction accuracy was found, both qualitatively and quantitively, with the relatively small number of training samples (n=10000). Localised under-prediction of negative pressures was observed because of the limited quantity of similar training examples.

Finally, in the third study (§6.3), in a development from the simpler interference cases in the previous chapter, a real district of London was used as context obstruction model to a realistic design principal model test case. Sensitivity analyses were conducted on the sampling of training wind speeds and geometric transformations of the design mesh in the context field for the test

feature wind speeds; finding that the prediction accuracy is relatively invariant to the test feature transformation. In general, prediction accuracy was again found to be acceptable, with localised over-prediction of both positive and negative pressure.

Chapter 7

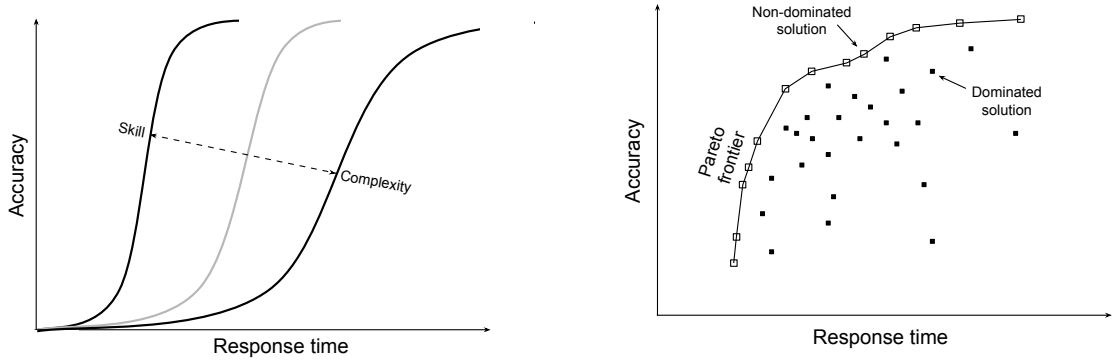
Discussion

This research explores a novel method for making fast-yet-accurate predictions of wind-induced pressure for the generative design of tall buildings. This involves a methodology where prediction error and time were calculated for cases of increasing levels of complexity: from validation of existing methods, through prediction for simple stand-alone and interference cases, to realistically complex building designs within a dense urban environment. This is achieved through the use of pre-computed sets of CFD simulations, from which vertex shape features and local fluid features were used to generate a reduced-order model.

In this discussion chapter, the results from the separate studies are compiled so that their time and accuracy can be compared against those of the various conventional analysis approaches.

7.1 Time Versus Accuracy

Speed-accuracy tradeoffs (SATs) were introduced in §3.1.2 (Figure 7.1a) and their slight reformation to Pareto frontiers in the time-accuracy objective domain §3.1.3 (Figure 7.1b). Escaping from the dependence between simulation time and accuracy in solver approximation is core to this thesis and specifically to the second hypothesis.



(a) Speed-accuracy tradeoff curve.

(b) Speed-accuracy objective space.

Figure 7.1: Representations of relationship between simulation speed and accuracy.

In the notional SAT depicted in Figure 7.1a, the curve can be transformed by either increasing the complexity of the task (a shift to the right) or increasing the level of expertise / skill (a shift to the left). For a fixed task therefore (here the task of assessing the wind-induced surface pressure on a tall building model), the tradeoff curve can be manipulated by incorporating knowledge-based systems; that is, skill can be a shortcut to achieving accuracy quickly. This concept is more easily represented and assessed by the Pareto frontier and the distinction between dominated / non-dominated solutions.

As mentioned, the relationship between analysis response time and accuracy of the presented solution approximation approach relative to existing solver approximation ones is necessary for validating the second hypothesis. In Chapter 4 the time and accuracy of the existing methods (FFD, RANS, LES, and wind-tunnel) are assessed. This provides a fundamental ground-truth against which to measure the success of any subsequent proposals; and suggests accepted tool specifications (time and accuracy) in practice, belonging to one of these approaches. In Chapters 5 and 6 the time and accuracy for the reduced-order model predictions of RANS and LES are assessed. Now in this section, these results are collected and combined.

7.1.1 CFD validation results

Reducing all of the validation studies to single metrics is clearly a broad generalisation in which the case variability is lost. However by doing so, a broader perspective can be achieved; that of positioning each approach globally and establishing the framework for assessing the thesis. The accuracy and time of the FFD, RANS, LES, and wind-tunnel are reported in Table 7.1.

Table 7.1: Mean validation errors and test times.

Approach	**Mean Absolute Relative Error** [%]	**Mean Absolute Relative Accuracy** [%]	**Mean Time** [s]
FFD	21.016	78.983	416.67
RANS	5.112	94.888	1703.4
LES	4.274	95.726	11640
WT	2.524	97.476	86400

Table 7.1 gives the *Mean Absolute Relative Error* [%], the *Mean Absolute Relative Accuracy* [%], and the *Mean Time* [s]. The *Mean Time* is simply the average over all the studies of that particular approach. The *Mean Absolute Relative Error* is averaged over all validation studies, the mean over all absolute errors relative to the ground-truth. Subsequently, the *Mean Absolute Relative Accuracy* is simply 100% - *Mean Absolute Relative Error*.

7.1.2 ROM sample-based results

Sample-based training errors are collected from the randomised test set m, where $m = D - n$, and for each study the total available data set size D varies. The sample-based test sets are larger and give a more uniformly randomised test of individual samples. They therefore give a good indication of the performance of the machine learning algorithm and training process. Model-based tests, although for smaller sets, represent a coherent test of a full model as would be found in practice, allowing for measurement of more realistic errors and prediction times. The latter is therefore used for assessing the hypotheses.

The errors of each of the studies sample-based tests are summarised in Table 7.2 and in Figures 7.2 and 7.3. Again, the minimum and maximum values shown in Figure 7.2 represent the errors found in only the worst-case individual samples or points from the large sets.

Table 7.2: Summary of sample-based errors.

Case	D	n	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$
Global	400	300	-11.417	14.666	3.960	4.284
Insolation	5000	50	-0.2415	0.2271	0.0235	0.033
Orientation	334339	10000	-47.090	32.981	1.471	1.929
Height	185155	10000	-18.985	19.307	0.878	1.102
Topology	166285	10000	-41.692	41.738	2.398	2.644
Super-formula	243440	10000	-42.284	51.890	3.269	3.565
Single/multi. int.	676395	10000	-43.144	45.893	1.017	1.615
Transient min.	853884	10000	-58.284	64.514	3.580	3.921
Transient max.	853884	10000	-42.180	46.666	2.720	2.947
Complex geometry	5723046	10000	-59.365	63.634	2.767	3.420
Complex interference	210000	10000	-51.906	40.115	1.217	1.756

In terms of the total prediction error range, the LES minimum peak pressure and the complex geometry studies have the greatest range; although all of the studies fit within the ±60% worst-case-sample error range. For the local feature studies, the mean absolute errors of all the studies are between 0.88 and 3.58%, with a standard deviation absolute error range of 1.10 to 3.92%.

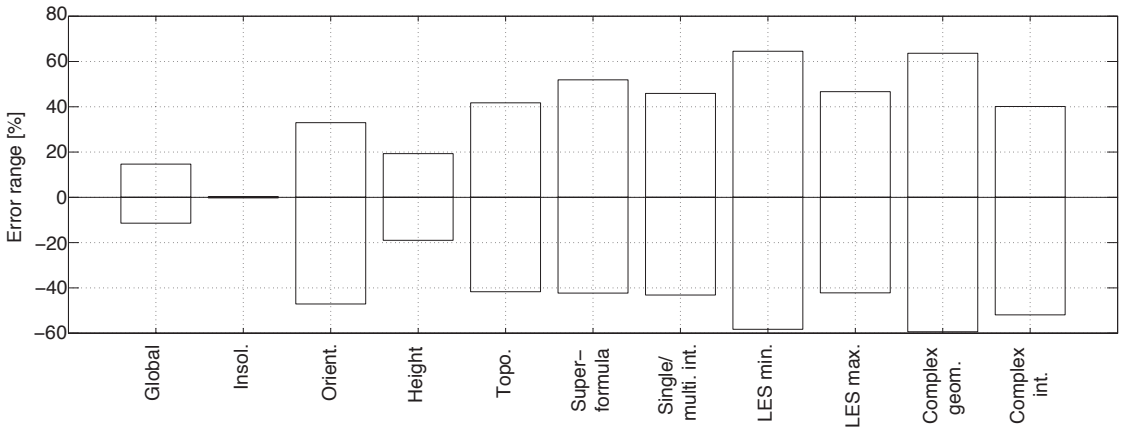


Figure 7.2: Sample-based error summary: $\delta_{min.}$ and $\delta_{max.}$.

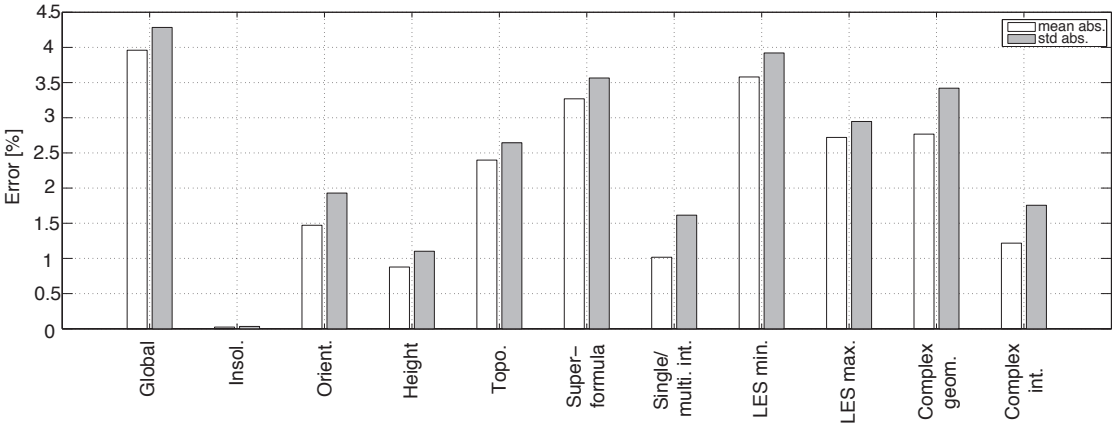
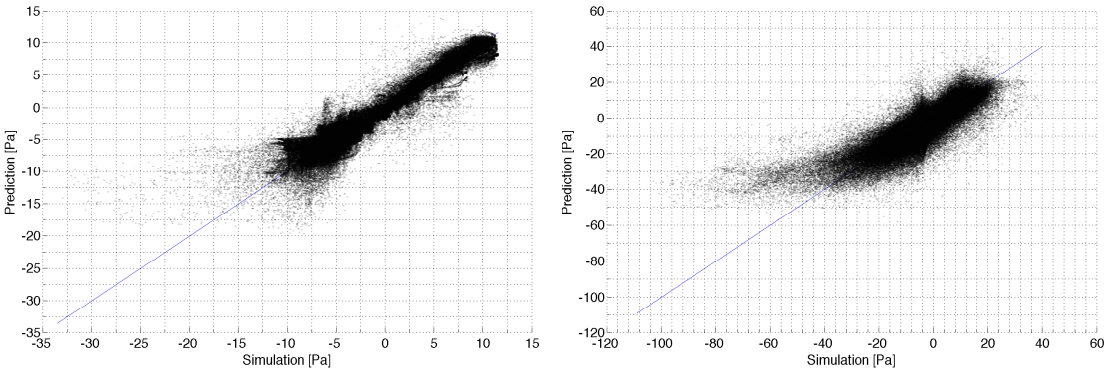


Figure 7.3: Sample-based error summary: $\overline{|\delta|}$ and $\sigma_{|\delta|}$.

7.1.3 ROM sample-based error distribution

The sample-based error distribution, in the form of simulated against predicted pressures, is given in Figure 7.4 for the RANS orientation interpolation case (Figure 7.4a, §5.3.8) and the complex geometry case (Figure 7.4b, §6.2). Ideally, the distribution should tend towards a $y = x$ linear trend, shown in blue. In both cases, a training set size of n=10000 is used. The distribution is typical for all the other studies, particularly in the deviation from the $y = x$ line at low pressures. For the two cases shown, the overall correlation of the data with $y = x$ has an r^2=0.9772 and r^2=0.9621 respectively.

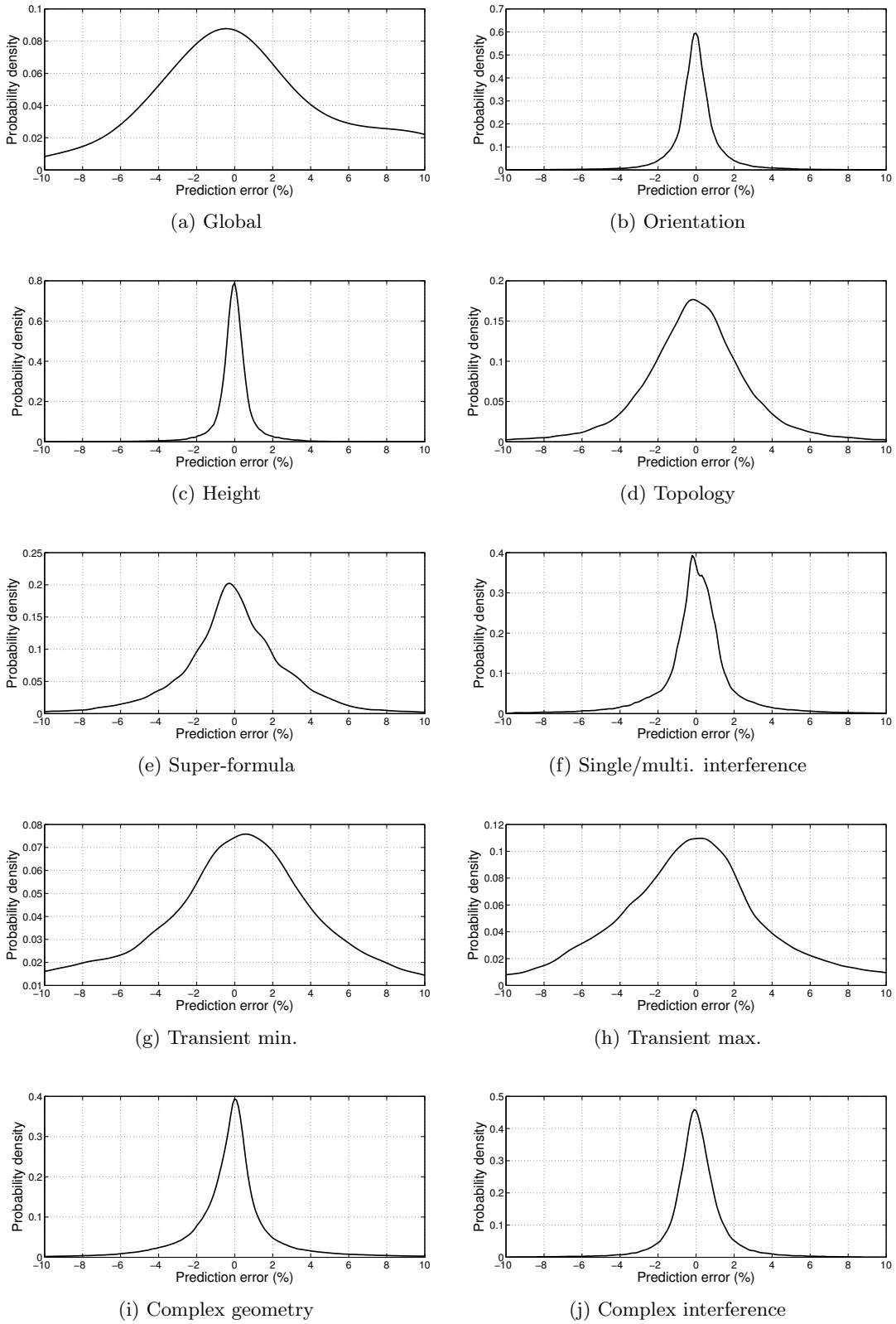
Although this low pressure region of deviation is significant, the number of samples is relatively low, in the order of hundreds compared to the 324339 (Figure 7.4a) or 5716831 (Figure 7.4b) samples plotted. This relatively consistent under-prediction of negative pressure values can be seen throughout on the model-based test plots, and may simply be due to a lower number of available training samples. Based on the over-training observed with the Random Forest algorithm (§5.3.7), it is likely that the under-prediction would be worse than for the ANN.



(a) Orientation (§5.3.8) (b) Complex geometry (§6.2)

Figure 7.4: Sample-based Y vs. Y' distribution.

For each study, the probability density of the sample-based errors is shown in Figure 7.5. Each uses the test set size $m = D - n$, where D varies but n is typically 10000. Figure 7.5 has a constrained error range of -10 to 10 [%] (x-axis), however the probability is scaled according to the case. The kernel sample density plots use a normal distribution and a kernel width of 0.1%.



(a) Global (b) Orientation

(c) Height (d) Topology

(e) Super-formula (f) Single/multi. interference

(g) Transient min. (h) Transient max.

(i) Complex geometry (j) Complex interference

Figure 7.5: Sample-based probability density distributions.

In all cases (Figure 7.6), the error probabilities can be seen to have a normal distribution with a mean of approximately zero. The largest distributions occur for the global (a), topology (d), and transient LES (g, h) cases.

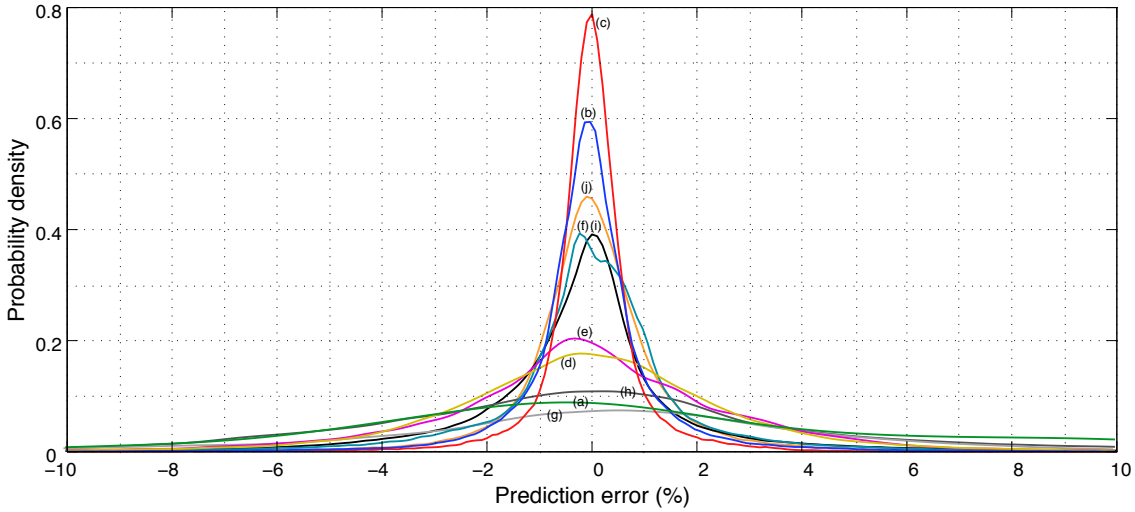


Figure 7.6: Sample-based probability density distributions: all cases.

7.1.4 ROM model-based results

The model-based results in the following tables summarise the errors and test times from each of the experiments described in the previous two chapters.

Table 7.3: RANS ROM model-based: summary of time vs. errors [%].

		Error, δ [%]				**Test time, t [s]**		
Case	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
Orientation								
5	16329	-28.925	15.251	1.198	1.423	454.607	0.0655	454.673
10	17354	-17.736	22.166	1.217	1.466	483.144	0.0615	483.205
20	18353	-17.432	20.370	1.053	1.333	510.957	0.0623	511.019
25	18966	-16.108	20.281	1.047	1.299	528.023	0.0615	528.084
35	19902	-14.273	36.288	0.943	1.451	554.081	0.0656	554.147
40	20107	-15.169	37.430	0.865	1.555	559.789	0.0608	559.850
50	20122	-14.624	38.790	0.871	1.638	560.206	0.0610	560.267
55	19912	-16.117	36.593	0.917	1.436	554.360	0.0625	554.422
65	18946	-14.548	19.698	1.137	1.429	527.466	0.0616	527.528
70	18302	-16.366	21.566	1.108	1.449	509.537	0.0642	509.601
80	17364	-27.819	27.931	1.154	1.492	483.422	0.0639	483.486
85	16352	-20.512	16.754	1.090	1.375	455.248	0.0625	455.310
Height								
15	3156	-12.066	13.619	1.632	1.950	87.863	0.0206	87.884
20	3885	-10.113	17.109	1.374	1.741	108.158	0.0250	108.183
30	5596	-8.401	13.426	0.991	1.264	155.793	0.0289	155.822
35	6328	-7.784	9.250	0.876	1.055	176.172	0.0315	176.203
45	8011	-7.266	11.855	0.764	0.839	223.026	0.0368	223.063
50	8909	-6.938	13.422	0.731	0.833	248.027	0.0392	248.066
60	10626	-7.979	13.131	0.673	0.762	295.828	0.0461	295.874
65	11351	-8.480	7.540	0.629	0.716	316.012	0.0487	316.061
75	13006	-11.144	11.955	0.635	0.748	362.087	0.0515	362.139
80	13964	-12.088	12.008	0.674	0.820	388.758	0.0513	388.809
90	15719	-16.269	12.529	0.705	0.886	437.617	0.0612	437.678
95	16379	-10.911	13.243	0.747	0.918	455.991	0.0634	456.055

(...continued on next page)

RANS ROM model-based: summary of time vs. errors [%] (Table continued.)

Topology								
4	18805	-19.014	32.532	4.941	4.806	523.531	0.0618	523.593
6	19075	-52.507	37.713	6.475	6.266	531.048	0.0624	531.110
8	18681	-30.023	19.520	3.448	3.893	520.079	0.0653	520.144
10	19245	-34.316	33.186	4.455	4.434	535.781	0.0614	535.842
Super-formula								
4,2,4,13	22204	-41.481	30.215	5.872	6.296	618.159	0.0789	618.238
6,20,7,18	30021	-52.015	49.347	8.140	8.315	835.785	0.0920	835.877
5,2,6,6	24054	-27.316	52.350	5.024	4.532	669.663	0.0816	669.745
4,1,7,8	31270	-33.097	46.344	6.768	5.893	870.557	0.0978	870.655
4,12,15,15	18013	-40.048	33.989	5.950	5.279	501.482	0.0702	501.552
12,15,20,3	17637	-42.660	34.478	7.423	6.003	491.014	0.0699	491.084
Interference								
Single	45093	-33.654	19.134	4.139	1.688	1255.389	0.1389	1255.528
Multi	45093	-28.175	23.306	6.744	3.985	1255.389	0.1389	1255.528
Complex Geometry								
Met Life	11439	-25.018	33.058	3.629	5.088	318.462	0.0468	318.509
Shard	20145	-21.899	13.849	2.392	2.498	560.837	0.0680	560.905
Sears	1629	-16.025	9.662	2.467	2.838	45.351	0.0230	45.374
Euston	1981	-19.770	12.314	3.669	4.059	55.151	0.0238	55.175
Taipei101	29091	-17.352	15.027	2.492	2.871	809.893	0.0898	809.983
Shanghai	10469	-23.578	25.268	4.440	4.840	291.457	0.0445	291.501
BankOfChina	528	-13.702	4.929	2.237	2.479	14.700	0.0203	14.720
Exchange	3931	-23.616	23.699	3.929	4.451	109.439	0.0286	109.468
Frankfurter	3688	-16.617	13.299	1.994	2.096	102.674	0.0280	102.702
Washington	1837	-19.099	13.380	2.766	2.225	51.142	0.0235	51.166
Complex Interference								
London City	14000	-55.798	18.963	5.821	9.103	389.760	0.0481	389.808

Table 7.4: LES ROM model-based: summary of time vs. errors [%].

		Error, δ [%]				**Test time, t [s]**		
Case	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
LES minimum								
5	41931	-49.490	40.744	3.563	3.511	1167.359	0.1207	1167.480
10	44537	-47.511	47.796	4.156	3.966	1239.910	0.1297	1240.040
20	46372	-41.917	50.139	4.140	4.555	1290.996	0.1342	1291.131
25	48447	-42.198	49.324	3.843	4.513	1348.764	0.1317	1348.896
35	50336	-35.198	51.855	3.862	3.924	1401.354	0.1407	1401.495
40	51177	-33.432	45.782	3.791	3.812	1424.768	0.1493	1424.917
50	51186	-35.918	49.404	3.729	3.759	1425.018	0.1407	1425.159
55	50314	-39.853	53.510	3.694	4.104	1400.742	0.1418	1400.884
65	48472	-49.209	51.235	3.608	4.428	1349.460	0.1363	1349.597
70	46318	-43.272	54.341	3.986	4.545	1289.493	0.1234	1289.617
80	44630	-50.082	53.672	4.236	4.165	1242.499	0.1219	1242.621
85	41855	-55.422	47.736	3.651	3.731	1165.243	0.1185	1165.362
LES maximum								
5	41931	-41.589	35.061	3.147	3.057	1167.359	0.1207	1167.48
10	44537	-41.894	36.982	3.130	3.219	1239.91	0.1297	1240.04
20	46372	-31.697	55.106	2.292	2.383	1290.996	0.1342	1291.131
25	48447	-32.387	38.345	2.100	2.364	1348.764	0.1317	1348.896
35	50336	-26.587	37.128	2.697	3.421	1401.354	0.1407	1401.495
40	51177	-33.482	40.911	2.662	3.088	1424.768	0.1493	1424.917
50	51186	-36.593	49.412	2.531	2.948	1425.018	0.1407	1425.159
55	50314	-27.422	34.843	2.786	3.465	1400.742	0.1418	1400.884
65	48472	-33.049	47.921	2.133	2.273	1349.46	0.1363	1349.597
70	46318	-33.061	43.425	2.357	2.372	1289.493	0.1234	1289.617
80	44630	-53.641	34.246	3.013	3.177	1242.499	0.1219	1242.621
85	41855	-50.042	36.758	2.949	2.973	1165.243	0.1185	1165.362

Table 7.5: Mean ROM errors and test times across all studies.

Approach	**Relative Mean Absolute Error [%]**	**Relative Mean Absolute Accuracy [%]**	**Mean Time [s]**
RANS ROM	7.841	92.159	442.46
LES ROM	7.526	92.474	1312.27

The average accuracies relative to the ground-truth and times of the RANS and LES ROM studies are given in Table 7.5. Again, the relative accuracies are simply calculated by addition of the ROM to either the RANS or LES error.

7.1.5 Pareto analysis

The accuracy and times from the validation studies are compiled in Figure 7.7, and the ROM studies in Figure 7.8. The errors are converted to accuracy by simply subtracting the error [%] from 100%, i.e. 100% - $\overline{|\delta|}$, so that the error and accuracy are inversely proportionate.

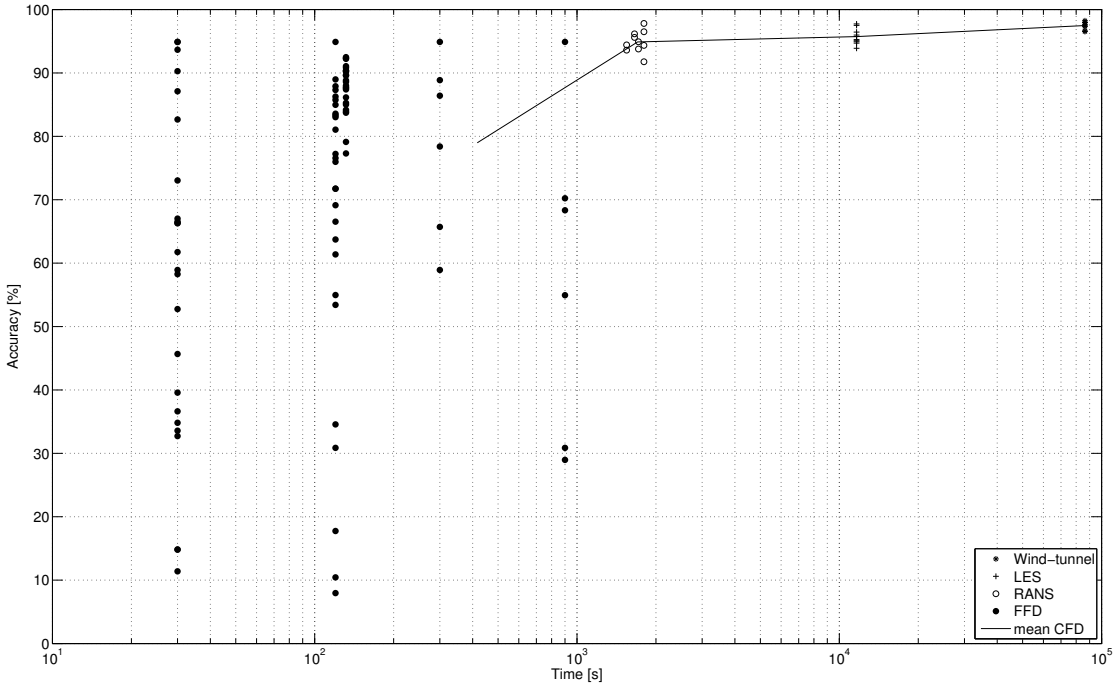


Figure 7.7: Collected FFD, CFD and wind-tunnel accuracy vs. time.

It is interesting to note that the mean times of the wind-tunnel, LES, RANS, and FFD are roughly equally spaced by orders of magnitude. This may be purely coincidental, or possibly due to an inadvertent tendency in development for each distinct analysis approach to distribute itself equidistantly between others. The variation in accuracy of the FFD is now clear over nearly the full 0 to 100% range; however the variation over time is partially distorted due to the log-time scale in Figure 7.7.

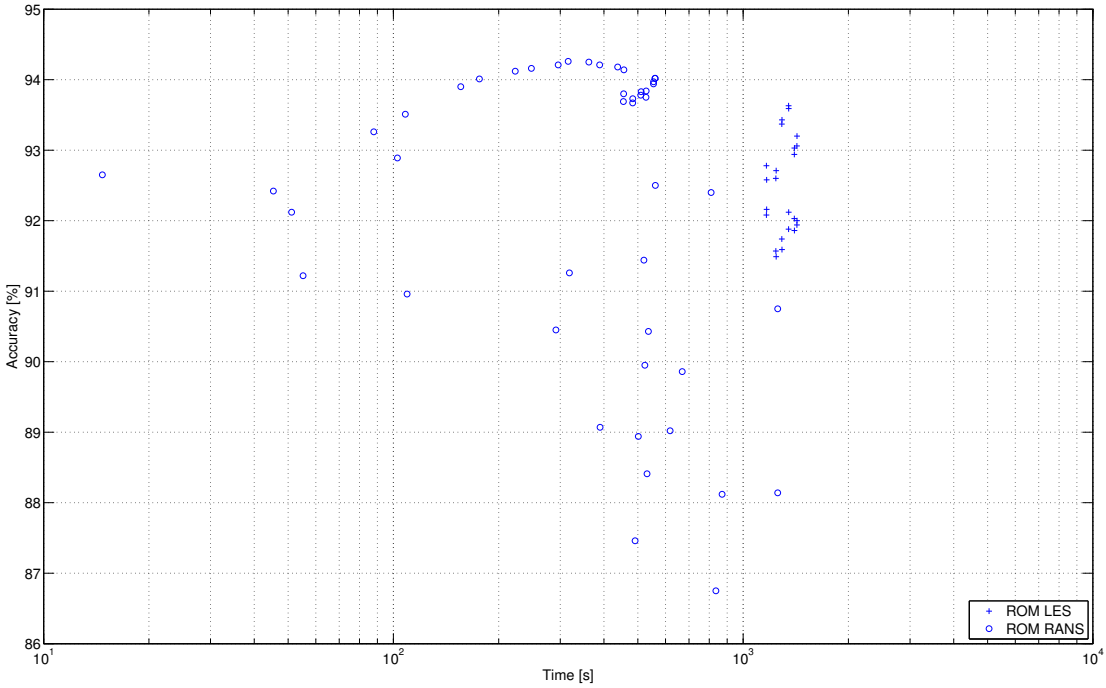


Figure 7.8: Collected ROM model-based prediction accuracy vs. time.

Subsequently, the reduced-order model studies can be positioned alongside the FFD and CFD validation studies. Time and accuracy data from all of the validation and experimental studies is compiled in Figures 7.9 and 7.10. A number of observations can be drawn from this; primarily, in response to the second hypothesis, that both the RANS and LES reduced-order model results alter the existing Pareto front as new non-dominated solutions.

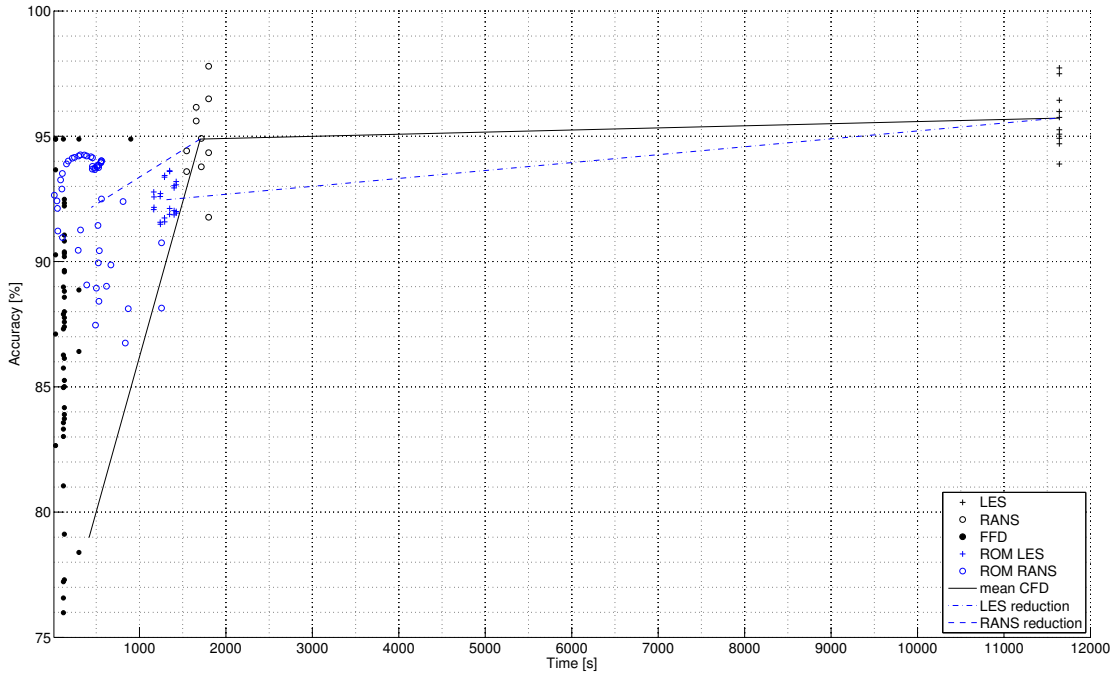


Figure 7.9: Time-accuracy of CFD and ROM results: 75-100% accuracy scale.

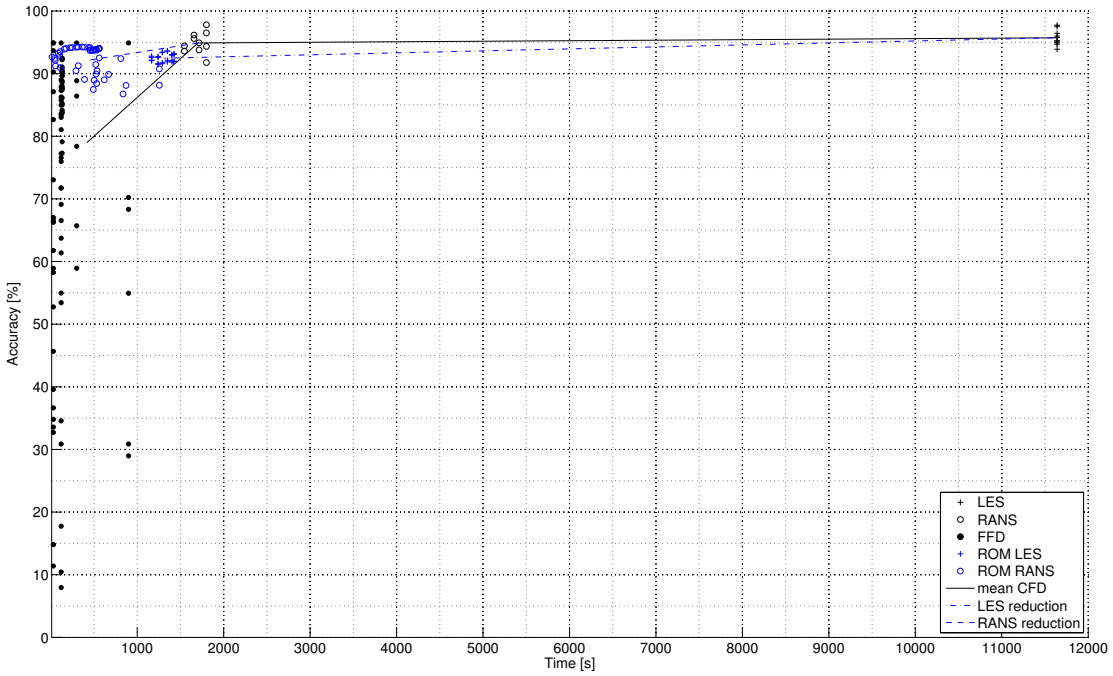


Figure 7.10: Time-accuracy of CFD and ROM results: 0-100% accuracy scale.

Across all studies and all model-based mean absolute errors, the mean RANS reduction in time and accuracy is 1260.94s (3.8-times faster) and 2.729%; and the LES reduction in time and accuracy is 10327.73s (8.9-times faster) and 3.252%. These reductions are depicted by the blue dotted lines in Figures 7.9 and 7.10 (these graphs are identical except for the y-axis Accuracy range).

It is clear that the LES ROM has greater benefits than the RANS ROM, indicating that the benefits increase with the intensity or cost of the simulation. In fact these results are conservative since as the LES requires a finer mesh than RANS, the number of samples in the test sets are also greater thus increasing the prediction test time. The RANS ROM is on average better than the FFD, however due to the result's variation in both there are few cases in which the FFD does appear to perform better in both time and accuracy.

7.2 Open-Source Learning

Outlined in Chapter 2 is a description of the analysis framework being developed by *Bentley*, with its local (GC) and online (analysis and optimisation) components (Figure 2.8). In this section, a developmental pathway for integration and extension of the approach is described.

The basic individual components of the process are shown in Figure 7.11 in columns, and the different implementation strategies in rows. There are two different implementation strategies as shown on the far right, divided into either *Training + Testing* for full control over the process or just *Testing* as a finalised product for the end-user. With development, the tool moves from the

Current scenario (top row) towards the fully developed *User (fast)* scenario (bottom row).

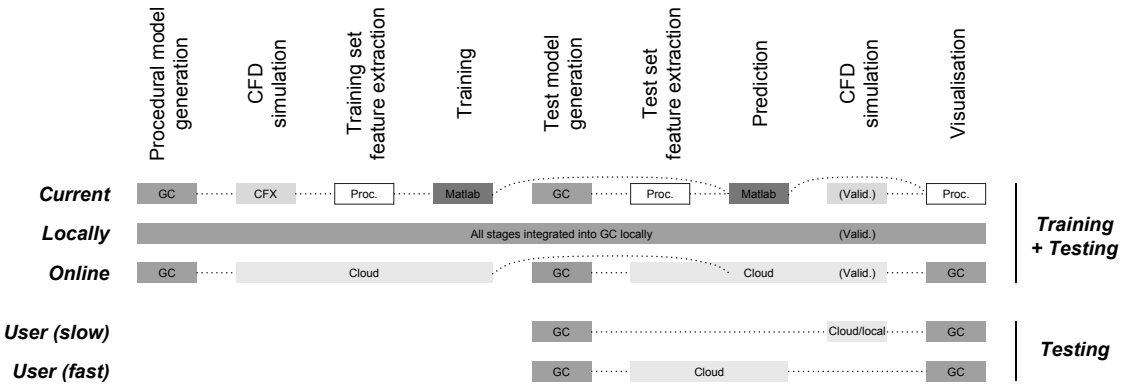


Figure 7.11: Stages of future prototype development.

Each of the individual developmental stages is explained below:

i) *Prototype:* The current prototypical state of the process is fragmented and not integrated with *GC*. The procedural models are generated in *GC*, CFD simulations are run in *ANSYS CFX*, feature calculation in *Processing*, and training and testing in *Matlab*.

ii) *Local:* This would be the first stage in commercial development, namely removing the need for *CFX*, *Matlab*, and *Processing*, bringing their functionality into *GC*. *Processing* is currently primarily used for generating the shape features from the surface meshes and visualising the end results, both of which can be handled relatively simply by *GC*'s embedded lightweight geometry and rendering capabilities. Replacing *Matlab* will involve writing a custom machine learning algorithm that can be trained on the extracted features and can store the reduced-order model (typically a simple data structure). *CFX* will need to be replaced by an in-house proprietary CFD solver of *Bentley*, but the integration of this process (meshing, boundary condition definition, solver convergence, etc.) will have to be developed at some level to maximise automation. Once these functions are fully streamlined within *GC*, the following stages of product development are relatively straight-forward.

iii) *Online:* Here there is a transition of the computationally demanding aspects of the process to the cloud, that is: evaluation of the training set; feature extraction; and the reduced-order model generation process itself. *GC* will remain the user-interface for the entire process.

iv) *User (Slow):* The first of two options proposed for the end-user, whereby no machine learning is involved but simply the CFD solver is integrated with *GC* either locally or online. Local and online CFD simulation simply necessitates submission of the model and certain boundary conditions, i.e. wind speed, direction, etc. This option is necessary if the user requires slower, but more accurate results. There is an opportunity to collect online simulation data and to add cumulatively to the training set for future learning.

v) *User (Fast):* The final stage in the product development of the end-user's tool is the simplest. *GC* is used as the interface to both create the design model and visualise the end results, with submission of the model to the cloud where the features will be generated, a prediction made from the learnt function, simply returning the response data.

For interoperability, the ROM which maps input features to output response can be exported intact as a single data structure. In the case of a random forest learning algorithm the output data structure is a decision tree or with an artificial neural network it is a matrix of neuron weights. These represent the absolute core of the whole process as they are the end product of the training phase. All of the back-end CFD simulation is distilled down into this specific matrix configuration which can be appended at any time with new input data.

The training and testing phases can therefore be separated neatly by extracting this data structure. The structures are also relatively compact: the ANN more so than the RF as its size is not dependent on the amount of training data, rather only the complexity of the problem (hidden layer size) or number of features used. In order to utilise this data structure in another program, all that is required is to present a new case with the correct number of input features, 'push' this through the structure, and collect the output response.

The learnt regression function represents a considerable amount of invested time; in the same way, anthropologically, as an experienced mind is typically more valuable than an inexperienced one as it has more time and experiences to configure itself to give better responses. This holds true for most problems in machine learning, whether it is the targeting of personalised advertising based on years worth of collected purchases or predictive search results trained on key words extracted from emails. There are many examples of websites that use machine learning to create tools that improve their performance over time, and essentially becomes more valuable as time goes by. The setup described here is not limited to use with CFD evaluation but could rather be seen as a model for any performance evaluation tool.

Accordingly, for *Bentley*, there is therefore a potential revenue associated with allowing access to it. For example, *PLP* or an end-user might pay a subscription to an online tool that provides light-weight performance evaluation, which removes the need for time-consuming and costly simulation. This is a similar approach to how architects currently involve consultants for guidance. With the option to run high-resolution simulation and optimisation over the cloud as the project develops, each time a model is submitted for explicit high-resolution simulation, the solution can be added to the database of features and incorporated into the training data for the machine learning. Data, or model, sharing could contribute towards credit for the user that could be redeemed in accessing the predictive tool in the future. This is described in Figure 7.12, representing both of the User scenarios in Figure 7.11.

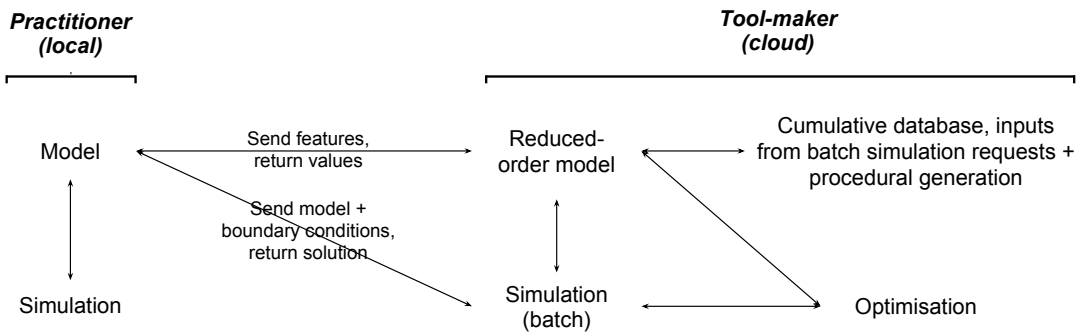


Figure 7.12: Proposed relationship between local / online simulation and prediction with *GC*.

The immediate benefit of learning is that the offline time taken to accumulate training data and learn from them can be separated from the online task response time. In this way using machine learning for predicting design performance, or function approximation for simulation, exploits the fact that the time previously required for evaluation can be back-loaded. This is ideally positioned where time and computational resources are abundant, i.e. pre-project, by a software provider, and on a distributed system.

Another benefit of shifting a costly process to a more opportune position is the freedom to dynamically update a centralised database with data of a controlled level of accuracy without affecting the response time. The issue of privacy concerns on sharing this kind of building performance data are avoided by the anonymity of the local sampling. That is, the vertex-based data supplied by the end-user can be randomised and therefore not linked back to an original design proposal.

7.3 Prototype Implementation

As a step towards product development, the neural network which has been trained in Matlab can be exported so as to be used externally for making predictions. It has been described how the trained neural network as a fixed entity represents a substantial amount of time and effort; i.e. procedural training geometry generation, training set CFD simulation and feature calculation, and the neural network training itself. After these stages, the network itself is a compact set of matrices which can be used for making predictions on new test cases.

The described network structure of X:H:Y=22:20:1 gives the following matrices of size [AxB]:

- input weights, **IW**=[20 x 22];
- input biases, **IB**=[20 x 1];
- hidden layer weights, **LW**=[1 x 20];
- output bias, **OB**=[1 x 1]

For a new test case, the matrix of input features is:

- input test model, **IT**=[22 x n].

Where n is the number of vertices on the test model mesh. For a new test prediction, Y', on model **IT**, the following two equations are required:

$$\mathbf{A} = tansig(\mathbf{IW} \times \mathbf{IT} + \mathbf{IB}) \tag{7.1}$$

$$Y' = tansig(\mathbf{LW} \times \mathbf{A} + \mathbf{OB}) \tag{7.2}$$

Where the tansig function (the hyperbolic tangent sigmoid transfer function) is:

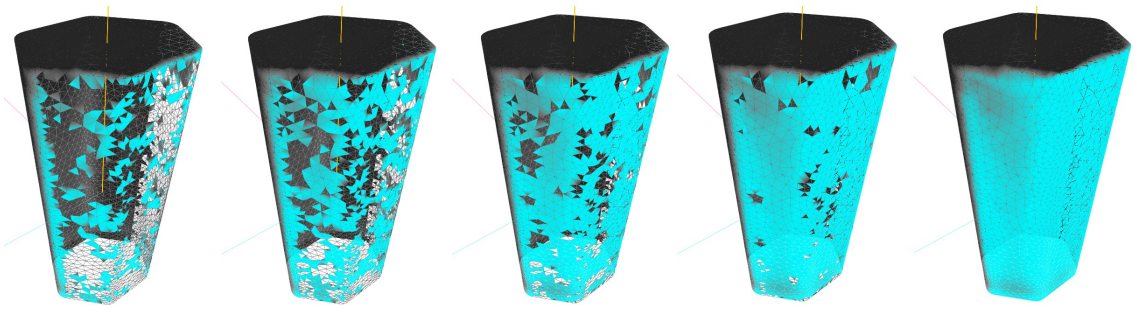
$$tansig(x) = \frac{2}{1 + exp(-2 \cdot x)} - 1 \tag{7.3}$$

This process is informally referred to as 'pushing' a new test through the network; in any software environment, it simply requires the coding of the matrix multiplications and additions, and the tansig function, as follows:

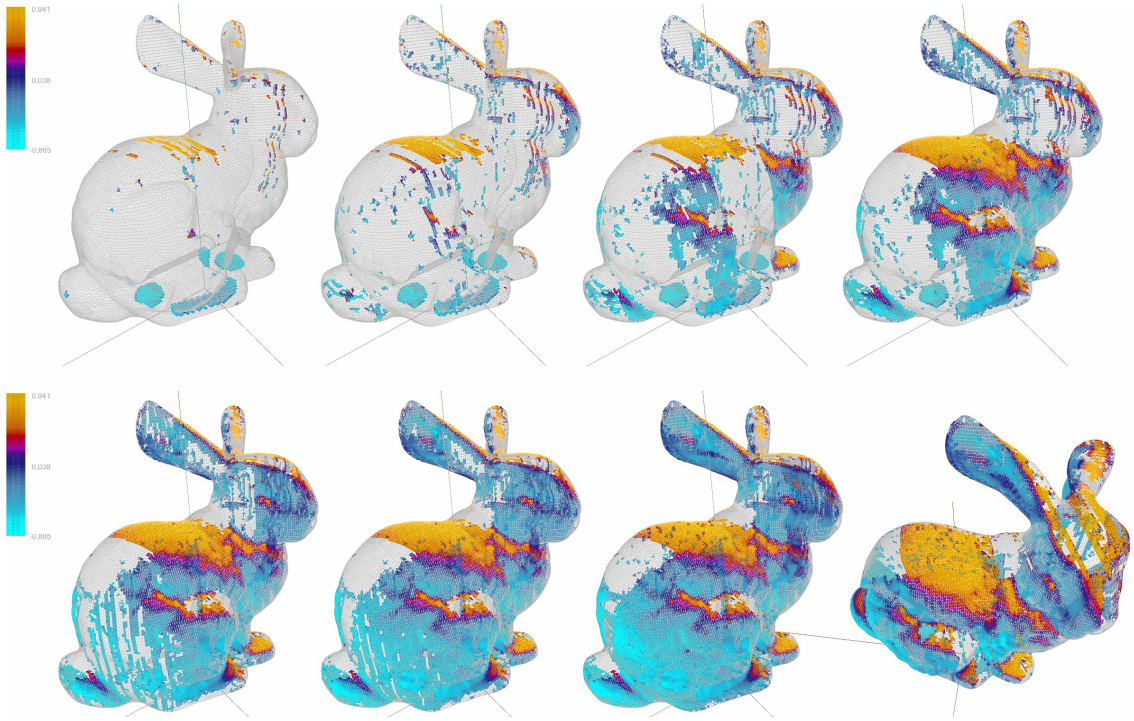
```
float pred(float IT) {
    for int i=0; i<20; i++; {
    float c=0;
        for int j=0; j<22; j++; {
            c+=IW[i][j]*IT[j];
           }
            A[i]=tansig(c+IB[i]);
      }
    float b=0;
    for int i=0; i<20; i++; {
        b+=(LW[i]*A[i]);
        }
    float Y=tansig(b+LB);
    return Y;
   }
```

At any point in time, the four neural network matrices (**IW**, **IB**, **LW**, and **OB**) can simply be replaced with more appropriate or improved ones. Since feature calculation is implemented in *Processing*, this can now be integrated with the final prediction. This means that a full test model mesh can be parsed, and then the features calculated and predictions made vertex-by-vertex (from left to right, Figures 7.13a and 7.13b show the progressive analysis and predictions of mesh vertices). Note that the pressure range legend scale must be updated for each additional vertex to find the minimum and maximum values.

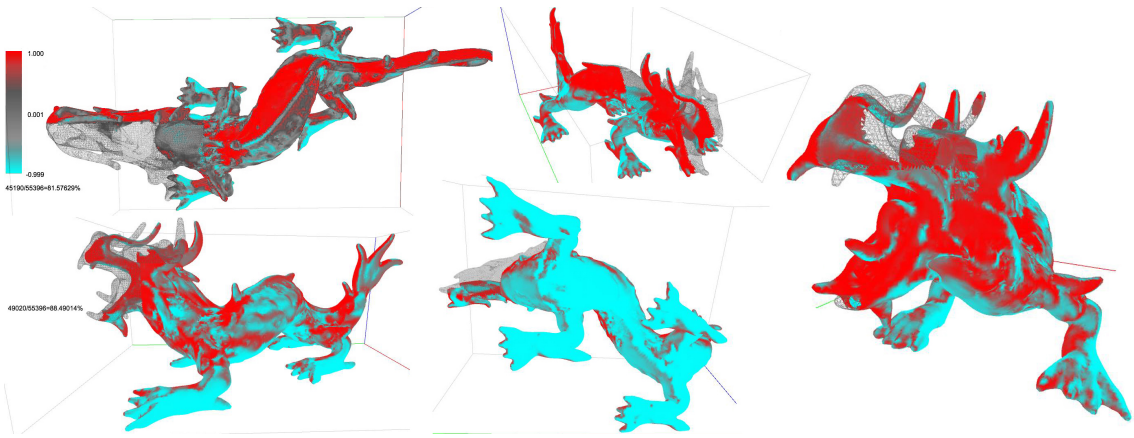
Integration in this environment allows for simple mesh transformations to be applied for new predictions also; such as rotation, scaling, and more free-form transformations. However, transitioning the code now to *GC* is relatively trivial, where the use of a more complex parametric model can be used for optimisation.



(a) Test case: procedural tall building.



(b) Test case: Stanford bunny.



(c) Test case: Chinese dragon.

Figure 7.13: Prototype implementation of simultaneous feature calculation and prediction.

7.4 Methodological Constraints

Field data: A key characteristic of the solution approximation approach is a reduction, where the full field data available through CFD simulation is reduced to the surface data of interest. As such, no direct information about the surrounding wind environment is conveyed as it would be with the full CFD. For certain cases this information is valuable; for instance, pedestrian comfort studies require data on the wind velocity at a horizontal plane above ground level in order to measure the change that a new design can have on an urban environment. The FFD is primarily focused on this field data with surface interactions as a secondary consideration; for the FFD the structured meshes give a poor surface representation and makes direct comparisons with CFD difficult.

Peak pressure: In the majority of cases steady-state RANS CFD was used, with the exception of the study into time-dependent peak pressures using LES. The prediction results are therefore not time-dependent peaks but time-averaged; a common approximation strategy in practice. This is a relatively significant simplification in structural engineering terms, but necessary to allow simulation of a large training set. In later project stages it is of concern to establish quantifiable peak values for the design, for which more accurate in-depth analyses are conducted, i.e. wind-tunnel or LES. However for the purposes of the thesis, the single LES study was sufficient to test the scalability of the reduced-order model to different basis data.

CFD Validation: It has been previously acknowledged that the purpose of the CFD and FFD validation (Chapter 4) was to create a contextual framework of their time and accuracy, and to create ground-truths, so as to position the subsequent predictions (Chapter 7). In order to achieve this general picture, i.e. arriving at single values representing the accuracies of a method for which the accuracies themselves are dependent on each specific case, the process has necessarily been reductive. In that respect, a variety of assessment strategies have been used, relying on multiple sources and metrics, which could potentially be more consistent with further work focussing specifically on this task. For instance, deficiencies in the physical accuracy of the FFD method meant that either field velocity (with errors which are conservative compared to pressures) or normalised surface pressure measurements (where there is no direct basis for physical comparison except for behavioural trends) were used, rather than surface pressure coefficients which are more consistent with other CFD validation literature.

Interference: In cases where interference must be considered, predictions require a simulation of the wider urban context (the obstruction model). Whilst the methodology only requires for this to be simulated once per wind direction, as opposed to for every change in the design model, it may still be seen as a hindrance. The contexts are typically dense urban environments that require detailed models, or reasonable simplifications, which take time for pre-processing and simulation. The amount of training data required for creating predictions with interference is much greater

than the context-free scenarios. Each training geometry must be evaluated for a range of wind speeds, increasing the offline process time without affecting the online prediction time.

Efficiency: The algorithm calculating the training and test shape features is not currently optimised for efficiency. The most costly part of the calculation lies in the local curvature analysis, where for every vertex the neighbouring vertices must be found and ordered by proximity (Appendix B.3). Consequently, the reported tests use a feature generation time of $0.02784 s/sample$ (about 36 samples per second) which is open for improvement.

Feature selection: Similarly for the input feature vector; a study into the sensitivity of the individual shape features is carried out showing the significance of each (§5.3.5), however there is no guarantee that the selection used is necessarily optimal. The local curvature analysis was calculated over five neighbourhood rings per vertex since the computational effort escalates quickly with the number of neighbourhoods. For the shape features, the optimal selection likely varies between geometry types and complexity, i.e. a simpler model would require less features to make an acceptable prediction. Which is also true for the interference cases, where the upstream distance at which the wind speed is taken for training and testing will vary between cases.

Chapter 8

Conclusion

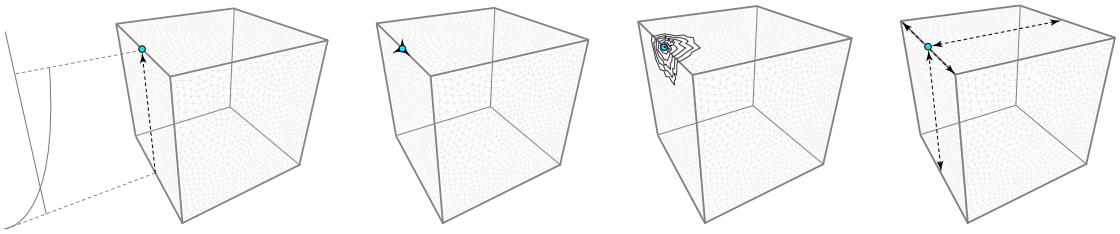
8.1 Hypotheses Response

The first hypothesis given in the introductory chapter (§1.3) is that: **i) Surface pressure simulated by CFD can be approximated by local geometric features.** Within the whole fluid domain, the nonlinearity and continuity of fluids causes a spatial and temporal dependence or sensitivity between all elements, which suggests that full domain simulation is necessary. The first hypothesis seeks to prove that through model reduction the simulated pressure can be approximated or learnt by local shape features. This implies that aspects of the simulated fluid domain, namely between the geometric description and pressure of a vertex, can be reduced to tractable relationships suitable for learning.

The evidence to support this hypothesis has been demonstrated in Chapters 5 and 6 through the series of experimental studies. Statistical error measurements and, perhaps more importantly from a practical perspective, the visualised surface pressure predictions and error distributions indicate the success of the proposed approach and solution approximation.

The degree to which the full domain is locally approximated is clearly dependent on the input feature vector or vertex shape definition used in the reduced-order model. These essentially identify geometric characteristics, $\{z,\mathbf{n},\mathbf{n}\sigma^{1-5},\mathbf{T}\}$, which correspond to the pressure response. The definition of each of these four types of feature is shown in Figure 8.1: vertex height (Figure 8.1a); vertex orientation (Figure 8.1b); vertex neighbourhood curvature (Figure 8.1c); and vertex relative position (Figure 8.1d).

The idea of the *local* geometric feature refers essentially to an *intrinsic* descriptive characteristic rather than an extrinsic property. As such, the locality of each feature varies between geometries and individual vertices; for instance, the locality of the $\mathbf{n}\sigma^{1-5}$ neighbourhood curvature is dependent on the mesh resolution. The objective of such a definition however, as opposed to the vertices' Cartesian co-ordinates $\{x, y, z\}$, is to associate a set of properties and a pressure value with each



(a) z (b) $\mathbf{n}\{x, y, z\}$ (c) $\mathbf{n}\sigma^{1-5}\{x, y, z\}$ (d) $\mathbf{T}\{x, y, z\}$

Figure 8.1: Locality of feature definitions.

vertex such that, by generalising these properties, learning can take place based on the assumption that vertices with similar properties will have similar pressure values.

The feature definition increases the dimensionality from the original Cartesian definition of $\mathbf{R}^3$ to $\mathbf{R}^{22}$, and in so doing increases the separation between samples. For instance, the Euclidean distance between two points, $\mathbf{d}(p, q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$, with unitary separation in each dimension, $(q_i - p_i) = 1$, the distance $\mathbf{d}$ increases from $\sqrt{3}$=1.73 to $\sqrt{22}$=4.69. The challenge this introduces of sampling a much greater space (the 'curse of dimensionalty') is countered by the increase in available samples from all vertices of a mesh.

The second hypothesis is that: **ii) Reduced-order model predictions dominate CFD simulations in both time and accuracy.** Existing fluid simulation methods tend towards being either fast-inaccurate or slow-accurate, whereas the reduced-order model can achieve fast-yet-accurate results. This is possible since the prediction accuracy and time of the reduced-order model are invariant, or independent, of basis simulation accuracy and time. Evidence for this is demonstrated by the positioning of the results as a non-dominated solution in the time-accuracy objective space and the subsequent alteration of the existing Pareto frontier.

Data from Chapters 4, 5, and 6 have been compiled in Chapter 7. In Figures 7.9 and 7.10, the time and accuracy results of the solver and solution approximation approaches are plotted together to show the existing and proposed Pareto frontiers. All of the ROM studies (blue markers) are positioned on the outside of the solver approximation Pareto frontier (black line). This is also simplified in Figure 8.2 showing just the mean accuracy and time of each approach. The second hypothesis is therefore supported by the data; that the ROM does present a novel non-dominated solution in this time-accuracy domain.

As mentioned, across all studies and all model-based mean absolute errors, the mean RANS ROM reduction in time and accuracy is 1260.94s (3.8-times faster) and 2.729%; and the LES ROM reduction in time and accuracy is 10327.73s (8.9-times faster) and 3.252%. These reductions are depicted by the blue dotted lines in Figure 8.2.

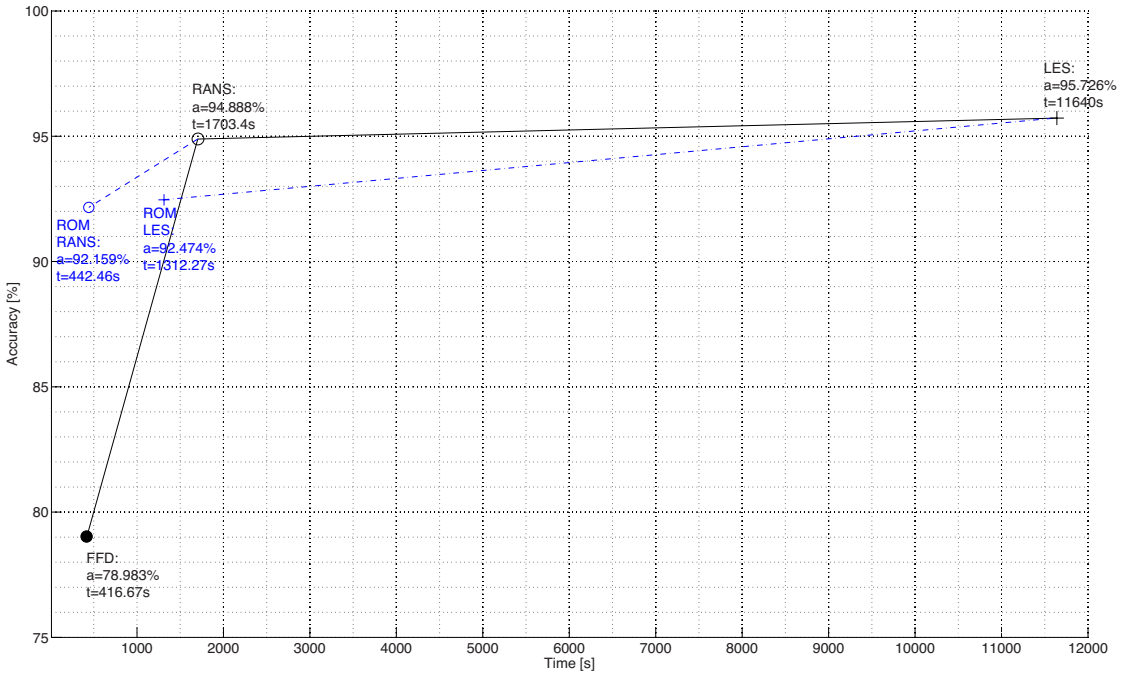


Figure 8.2: Average time (t) and accuracy (a) of CFD and ROM results.

8.2 Key Findings

This thesis investigates the use of reduced-order models to approximate computational fluid dynamic (CFD) simulations of the wind pressure on tall buildings. The results of the studies support two hypotheses: firstly, that wind-induced surface pressure on tall buildings, conventionally simulated with CFD, can be approximated for speed gains by local shape analysis; and secondly, that the methodology allows for a novel non-dominated approach in the time-accuracy domain, therefore altering the existing Pareto frontier of available analysis techniques. That is, response time was less than RANS and LES CFD, two methods accepted for their accuracy, whilst maintaining an accuracy greater than FFD, a method accepted for its speed. The reduced-order model can achieve fast-yet-accurate results, since prediction accuracy and time are invariant, or independent, of basis simulation accuracy and time; being instead solely a function of the reduced-order model performance and the geometric complexity or number of test mesh vertices.

Within existing research it was observed that trends in CFD, and other types of simulation, tend either towards accuracy *or* speed but rarely consider both. The prevalent notion in the literature of a speed-accuracy tradeoff is flawed in that it does not consider the issue as a multi-objective problem, therefore neglecting the idea of dominated approaches which are suboptimal with regards to both time and accuracy. Rephrasing this as a multi-objective problem adds clarity and allows for new methods to be judged by their effect on the existing Pareto set. The proposed model reduction methodology changes the existing tendency in a move towards a fast-yet-accurate analysis tool.

Existing work on CFD approximation has used either a global top-down approach, especially in the case of urban wind interference generalisation, or a local approach with non-unique spatial positions. In global approaches, single metrics are used to represent the overall performance of the design, such as the Strouhal number or peak pressure. The adoption of a global approach limits the flexibility by imposing a top-down parametric representation which constrains the potential of the generalisation; whilst the second only allows for interpolation within the single coordinate system and/or varying boundary conditions. The studies here on inclusion of interference have shown that the reduced-order model is capable of also taking into consideration local field conditions; and so greatly extending the potential of an alternative non-global approach to interference generalisation.

8.3 Contribution

An examination of the contribution of the presented work is discussed from three perspectives: i) designers or end-users such as *PLP*; ii) software developers such as *Bentley*; and iii) the broader research field of computational wind engineering (CWE). The benefits to the software end-user and developer have already been described in Chapter 2 and §7.2, in terms of developing this prototype approach into a usable tool.

The end-user provided the motivation for this thesis with the seemingly simple requirement for a *fast yet accurate* analysis tool. Accordingly, this has been achieved within the extents outlined in the previous chapter, suggesting through prototypes a new method for wind analysis in the generative design of tall buildings. The secondary advantage of the approach, which was not prescribed in the initial aims, is in the reduced level of expertise required as compared to typical CFD software. The reduced-order model simplifies the problem in that the end-user is removed from the technicalities of the simulation, for better or worse, concerning themselves solely with providing geometry.

The developer provides the context in which the motivation is framed, namely an existing parametric design software *GenerativeComponents* with its associated nascent analysis framework. Within this framework there is a place for conventional CFD analysis and an approximative reduced-order model system. As outlined previously these components can work in harmony: consisting of a CFD solver, the ROM, *GC*, an optimisation algorithm, and ultimately an online repository of training data.

Contributions are primarily made to two active sub-fields of computational wind engineering: CFD model reduction; and urban wind interference. In the first it was demonstrated that local shape characteristics of vertices along with their simulated pressures can be used successfully for generating reduced-order models with an ANN. This was assessed relative to existing conventional

analysis methods, namely RANS, LES, and wind-tunnel in the time-accuracy domain, with success suggested by alteration of the Pareto frontier. Secondly, urban wind interference is currently relying on global parameters for generalisation. The approach presented here is a markedly different alternative which warrants further investigation by the community.

8.4 Recommendations

The methodology presented in this thesis is prototypical in the sense that the input feature vector (the local description of a vertex's shape), the algorithms used to calculate them, and the reduced-order model generation most likely have suboptimal efficiency. The response time can be further improved by increasing the efficiency of the calculation method for the input vector; whilst further study on the selection, and problem-specific automatic selection, of the feature set will most likely improve the accuracy of the predictions. For instance, there is a vast amount of research on feature detection for image processing which can be explored for use on 3-d geometry.

The greatest time improvements were seen on the LES cases, indicating that the benefits of the approach increase as the training simulation time increases. As computing power increases LES, and perhaps in turn DNS, will replace RANS as the standard solver approach, producing generally more accurate results. However the improvements in speed brought by increased computer power are typically offset by increases in resolution (finer meshes, transient simulations, and shorter time-steps). Therefore, further investigation into the use of LES and DNS as the basis for the reduced-order model would be of great interest.

The local shape feature reduced-order model is applicable to other analysis tools and fields; namely where there is a design process of shape-based performance in a fluid environment. Benefits can also be found in integration with structural analysis, such as for the naval, aeronautics, aerospace, or automotive industries.

Finally, the integration of the methodology with an online analysis framework and an optimisation routine, as suggested, would be the final stage in the development towards a functioning tool. Further questions remain about the impacts on prediction accuracy of an evolving training set and reduced-order model, and how this process can be managed to maintain a good sample distribution. The obstacles to achieving this generally relate to the implementation, however the concept of an evolving online data set and reduced-order model is a promising area for further research.

References

Addington, M. D. (2003). New Perspectives on Computational Fluid Dynamics Simulation. In A. M. Malkawi & G. Augenbroe (Eds.), Advanced Building Simulation (First ed., pp. 141–158). New York and London: Spon Press, Taylor and Francis Group.

Agrawal, N., Mittal, A. K., & Gupta, V. K. (2012). Along-Wind Interference Effects on Tall Buildings. In National Conference on Wind Engineering, India (pp. 193–204).

Aish, R., & Woodbury, R. (2005). Multi-level Interaction in Parametric Design. In Smart Graphics (pp. 151–162). Springer.

Allaire, D., He, Q., Deyst, J., & Willcox, K. (2012). An Information-Theoretic Metric of System Complexity with Application to Engineering System Design. *Journal of Mechanical Design*, *134*.

Allaire, D., & Willcox, K. (2013). A Mathematical and Computational Framework for Multifidelity Design and Analysis with Computer Models. *International Journal for Uncertainty Quantification*, 1–30.

ANSYS. (2009). CFX-Solver Theory Guide (Tech. Rep.). Canonsburg, PA: ANSYS Inc.

ANSYS. (2014). CFX v13.0. Retrieved 05.02.2014, from www.ansys.com

Apostolopoulos, S. (n.d.). OpenGL Toolkit. Retrieved 17.08.2012, from http://www.opentk.com/

Augenbroe, G. (2003). Trends in Building Simulation. In A. M. Malkawi & G. Augenbroe (Eds.), Advanced Building Simulation (First ed., pp. 4–24). New York and London: Spon Press, Taylor and Francis Group.

Bailey, P. A., & Kwok, K. C. S. (1985). Interference Excitation of Twin Tall Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *21*, 323–338.

Baker, C. J. (2007). Wind Engineering - Past, Present and Future. *Journal of Wind Engineering and Industrial Aerodynamics*, *95*(9-11), 843–870.

Baker, W. F., & Pawlikowski, J. J. (2011). The World's Tallest Building. Retrieved 01.06.2011, from www.structuremag.org/article.aspx?articleID=1276

Basak, D., Pal, S., & Patranabis, D. C. (2007). Support Vector Regression. *Neural Information Processing*, *11*(10), 203–224.

Beaumont, C. N., Goodman, A. A., Kendrew, S., Williams, J. P., & Simpson, R. (2014). The Milky Way Project: Leveraging Citizen Science and Machine Learning to Detect Interstellar Bubbles. *Astrophysical Journal Supplement*.

Bentley Systems. (2013). GenerativeComponents v08.11.08.296. Retrieved 01.02.2014, from www.bentley.com

Bitsuamlak, G. (2006). Application of Computational Wind Engineering: A Practical Perspective. In Third National Conference in Wind Engineering (pp. 1–19).

Bitsuamlak, G., Dagnew, A. K., & Gan Chowdhury, A. (2008). Computational Blockage and Wind Simulator Proximity Effects Assessment for a new Full-scale Testing Facility. In 4th International Conference, Advances on Wind and Structures (AWAS08), Jeju, Korea.

Blocken, B. (2014). 50 years of Computational Wind Engineering: Past, Present and Future. *Journal of Wind Engineering and Industrial Aerodynamics*, *129*, 69–102.

Blottner, F. G. (1990). Accurate Navier-Stokes Results for the Hypersonic Flow over a Spherical Nosetip. *Journal of Spacecraft and Rockets*, *27*(2), 113–122.

Boehm, B. W. (1981). Software Engineering Economics. In Pioneers and Their Contributions to Software Engineering (pp. 99–150). Springer Berlin Heidelberg.

Brabanter, K. D., Karsmakers, P., Ojeda, F., Alzate, C., Brabanter, J. D., Pelckmans, K., ... Suykens, J. A. K. (2011). LS-SVMlab Toolbox Users Guide (Tech. Rep. No. August). Leuven-Heverlee, Belgium: Katholieke Universiteit Leuven.

Breiman, L. (2001). Random Forests. *Machine Learning*, *2001*(45), 5–32.

Bui-Thanh, T., Willcox, K., & Ghattas, O. (2008). Parametric Reduced-Order Models for Probabilistic Analysis of Unsteady Aerodynamic Applications. *AIAA*, *46*(10).

Burns, J. G. (2005). Impulsive Bees Forage Better: the Advantage of Quick, Sometimes Inaccurate Foraging Decisions. *Animal Behaviour*, *70*(6), e1–e5.

Campbell-Dollaghan, K. (2013). How This 116-Story Skyscraper Will 'Confuse' the Wind. Retrieved 01.05.2013, from www.gizmodo.com/how-this-116-story-skyscraper-will-confuse-the-wind-508206826

Cermak, J. E. (1976). Aerodynamics of Buildings. *Annual Review of Fluid Mechanics*, *8*(75).

Chen, Q., & Zhai, Z. (2003). The Use of Computational Fluid Dynamics Tools for Indoor Environmental Design. In A. M. Malkawi & G. Augenbroe (Eds.), Advanced Building Simulation (First ed., pp. 119–140). New York and London: Spon Press, Taylor and Francis Group.

Chiba, K., Jeong, S., Obayashi, S., & Morino, H. (2005). Data Mining for Multidisciplinary Design Space of Regional-Jet Wing. *IEEE*, 2333–2340.

Chittka, L., Dyer, A. G., Bock, F., & Dornhaus, A. (2003). Bees Trade Off Foraging Speed for Accuracy. *Nature*, *424.6947*(2003), 388–388.

Chittka, L., Skorupski, P., & Raine, N. E. (2009). Speed-Accuracy Tradeoffs in Animal Decision Making. *Trends in Ecology & Evolution*, *24*(7), 400–407.

Chronis, A., Tsigkari, M., Davis, A., & Aish, F. (2012). Design Systems, Ecology and Time. In Proceedings of ACADIA12: Synthetic Digital Ecologies. San Francisco, CA.

Chronis, A., Turner, A., & Tsigkari, M. (2011). Generative Fluid Dynamics: Integration of Fast Fluid Dynamics and Genetic Algorithms for Wind Loading Optimization of a Free Form Surface. In Proceedings of SimAUD SCS SpringSim'11 (pp. 79–86).

Cipriano, G., Phillips, G. N., & Gleicher, M. (2009). Multi-Scale Surface Descriptors. *IEEE Transactions on Visualization and Computer Graphics*, *15*(6), 1201–8.

Cochran, L., & Derickson, R. (2011). A Physical Modeler's View of Computational Wind Engineering. *Journal of Wind Engineering and Industrial Aerodynamics*, *99*(4), 139–153.

Colombo, E., Grassano, A., & Perotti, F. (2006). Comparison of Numerical and Experimental Simulations used to Investigate the Wind-Structure Interaction of a Skyscraper. In The Fourth International Symposium on Computational Wind Engineering (pp. 397–400).

Cortes, C., & Vapnik, V. N. (1995). Support-Vector Networks. *Machine Learning*, *20*(1995), 273–297.

Criminisi, A., Shotton, J., & Konukoglu, E. (2011). Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. *Foundations and Trends in Computer Graphics and Vision*, *7*(2-3), 81–227.

CTBUH. (2009). Tall Buildings in Numbers. Tall Buildings and Embodied Energy. *CTBUH Journal*, *2009*(III), 50–51.

CTBUH. (2010). Tall Buildings in Numbers. The Economics of High-rise. *CTBUH Journal*, *2010*(III), 44–45.

CTBUH. (2011). The Tallest 20 in 2020: Entering the Era of the Megatall (Tech. Rep.). Chicago: Council on Tall Buildings and Urban Habitat.

CTBUH. (2012). Tall Buildings in Numbers. Canada Rising. *CTBUH Journal*, *2012*(IV).

CTBUH. (2013). Criteria for the Defining and Measuring of Tall Buildings (Tech. Rep.). Chicago: Council on Tall Buildings and Urban Habitat.

CTBUH. (2014). The Global Tall Building Database of the Council on Tall Buildings and Urban Habitat. Retrieved 25.07.2013, from www.skyscrapercenter.com

Dagnew, A. K., Bitsuamlak, G., & Merrick, R. (2009). Computational Evaluation of Wind Pressures on Tall Buildings. In 11th Americas Conference on Wind Engineering (pp. 1–17).

Darom, T., & Keller, Y. (2012). Scale-Invariant Features for 3-D Mesh Models. *IEEE Transactions on Image Processing: a publication of the IEEE Signal Processing Society*, *21*(5), 2758–69.

Davenport, A. (1995). How can we simplify and generalize wind loads? *Journal of Wind Engineering and Industrial Aerodynamics*, *54/55*, 657–669.

Davidson, S. (2013). Grasshopper. Retrieved 01.02.2014, from www.grasshopper3d.com

Davoudabadi, P. (2012). The Most Accurate and Advanced Turbulence Capabilities. ANSYS.

Deardorff, J. W. (1970). A Numerical Study of Three-dimensional Turbulent Channel Flow at Large Reynolds Numbers. *Journal of Fluid Mechanics*, *41*(2), 453–480.

Degroote, J., Vierendeels, J., & Willcox, K. (2010). Interpolation Among Reduced-Order Matrices to Obtain Parameterized Models for Design, Optimization and Probabilistic Analysis. *International Journal for Numerical Methods in Fluids*, *2010*(63), 207–230.

Delaunay, D., Lakehal, D., & Pierrat, D. (1995). Numerical Approach for Wind Loads Prediction on Buildings and Structures. *Journal of Wind Engineering and Industrial Aerodynamics*, *57*(2-3), 307–321.

Demuth, H., & Beale, M. (2002). Neural Network Toolbox User's Guide V4 (Tech. Rep.). Natick, MA.: The MathWorks.

de Wilde, P. J. C. J., van der Voorden, M., Brouwer, J., Augenbroe, G., & Kaan, H. (2001). The Need for Computational Support in Energy-Efficient Design Projects in the Netherlands. In Seventh International IBPSA Conference (pp. 513–520). Rio de Janeiro, Brazil: Building Simulation.

Dong, C.-S., & Wang, G.-Z. (2005). Curvatures Estimation on Triangular Mesh. *Journal of Zhejiang University SCIENCE*, *6*(Suppl. I), 128–136.

Duffy, A. H. B. (1997). The 'What' and 'How' of Learning in Design. *IEEE Expert*, *12*(3), 71–76.

Duvigneau, R., & Visonneau, M. (2003). Shape Optimisation Strategies for Complex Applications in Computational Fluid Dynamics (Tech. Rep.). Nantes: Ecole Centrale de Nantes.

Eggenspieler, G., & Menter, F. R. (2012). Turbulence Modeling. ANSYS.

Elcott, S., Tong, Y., Kanso, E., Schroder, P., & Desbrun, M. (2007). Stable, Circulation-Preserving, Simplicial Fluids. *ACM Transactions on Graphics*, *26*(1).

Elliott, J., & Peraire, J. (1996). Progress Towards a 3D Aerodynamic Shape Optimisation Tool for the Compressible, High-Re Navier-Stokes Equations Discretized on Unstructured Meshes (Tech. Rep.). Boston: MIT Department of Aeronautics and Astronautics.

English, E. C., & Fricke, F. R. (1999). The Interference Index and its Prediction using a Neural Network Analysis of Wind-Tunnel Data. *Journal of Wind Engineering and Industrial Aerodynamics*, *83*, 567–575.

Esri. (2014). CityEngine. Retrieved 15.01.2014, from www.resources.arcgis.com/en/communities/city-engine

Fedkiw, R., Stam, J., & Jensen, H. W. (2001). Visual Simulation of Smoke. In SIGGRAPH.

Feldman, B. E., O'Brien, J. F., & Klingner, B. M. (2005). Animating Gases with Hybrid Meshes. *ACM Transactions on Graphics*, *24*(3).

Feng, X., Xia, K., Chen, Z., Tong, Y., & Wei, G.-W. (2013). Multiscale Geometric Modeling of Macromolecules II: Lagrangian Representation. *Journal of computational chemistry*, *34*(24), 2100–20.

Ford, D. N. (1994). Ferrybridge Cooling Towers Collapse. *When Technology Fails: Significant Technological Disasters, Accidents, and Failures of the Twentieth Century*.

Frey, P., & George, P.-L. (2010). Mesh Generation. John Wiley and Sons.

Friedman, J. H. (1990). Multivariate Adaptive Regression Splines. *The Annals of Statistics*(August).

Fu, C. L., Lee, S. M., & Cheng, C. M. (2006). Validation of CFD Simulations on the Wind Loads for Tall Buildings Preliminary Design. In The Fourth International Symposium on Computational Wind Engineering (pp. 369–372).

Giannakoglou, K. C. (2002). Design of Optimal Aerodynamic Shapes using Stochastic Optimization Methods and Computational Intelligence. *Progress in Aerospace Sciences*, *38*(1), 43–76.

Gielis, J. (2003). A Generic Geometric Transformation that Unifies a Wide Range of Natural and Abstract Shapes. *American Journal of Botany*, *90*(3), 333–338.

Gomes, M. G., Rodrigues, A. M., & Mendes, P. (2005). Experimental and Numerical Study of Wind Pressures on Irregular-Plan Shapes. *Journal of Wind Engineering and Industrial Aerodynamics*, *93*(10), 741–756.

Gosnell, M. R., Woodley, R. S., & Gorrell, S. E. (2012). Results of Mining Data Features During Computational Fluid Dynamics Simulations (Tech. Rep.). Provo, UT: Brigham Young University.

Graening, L., Menzel, S., Hasenjäger, M., Bihrer, T., Olhofer, M., & Sendhoff, B. (2008). Knowledge Extraction from Aerodynamic Design Data and its Application to 3D Turbine Blade Geometries. *Journal of Mathematical Modelling and Algorithms*, *7*(4), 329–350.

Graening, L., & Sendhoff, B. (2014). Shape Mining: A Holistic Data Mining Approach for Engineering Design. *Advanced Engineering Informatics*, *28*(2), 166–185.

Gu, M. (2009). Study on Wind Loads and Responses of Tall Buildings and Structures. In 7th Asia-Pacific Conference on Wind Engineering.

Gu, M., & Xie, Z.-N. (2011). Interference Effects of Two and Three Super-Tall Buildings under Wind Action. *Acta Mechanica Sinica*, *27*(5), 687–696.

Gunn, S. R. (2010). Support Vector Machines for Classification and Regression. (Vol. 135; Tech. Rep. No. 2). Southampton: University of Southampton.

Gürsel Dino, I. (2012). Creative Design Exploration by Parametric Generative Systems in Architecture. *METU JFA*, *1*(29:1), 207–224.

Hanna, S. (2007). Inductive Machine Learning of Optimal Modular Structures: Estimating Solutions using Support Vector Machines. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *21*(04), 351–366.

Hanna, S. (2011). Addressing Complex Design Problems through Inductive Learning. Unpublished doctoral dissertation, University College London.

Haykin, S. (1999). Neural Networks: A Comprehensive Foundation (2nd ed.). New Jersey: Prentice Hall.

Holland, J. H., Holyoak, K. J., Nisbett, R. E., & Thagard, P. R. (1987). Induction. The Massachusetts Institute of Technology.

Holscher, N., & Niemann, H. J. (1998). Towards Quality Assurance for Wind-Tunnel Tests: A Comparative Testing Program of the Windtechnologische Gesellschaft. *Journal of Wind Engineering and Industrial Aerodynamics*, *74*, 599–608.

Hsu, S. A., Meindl, E. A., & Gilhousen, D. B. (1994). Determining the Power-law Wind-profile Exponent under Near-neutral Stability Conditions at Sea. *Journal of Applied Meteorology*, *33*(1994), 757–772.

Hubeli, A., & Gross, M. (2001). Multiresolution Feature Extraction for Unstructured Meshes. In Proceedings of the conference on Visualization'01. IEEE Computer Society (pp. 287–294).

Hubeli, A., Meyer, K., & Gross, M. (2000). Mesh Edge Detection (Tech. Rep.). Zurich: Swiss Federal Institute of Technology.

Hume, D. (1896). A Treatise of Human Nature. Dover Philosophical Classics.

Huyse, L. (2001). Free-form Airfoil Shape Optimization Under Uncertainty Using Maximum Expected Value and Second-order Second-moment Strategies (Tech. Rep.). Hampton, Virginia: ICASE, NASA Langley Research Center.

Huyse, L., & Lewis, R. M. (2001). Aerodynamic Shape Optimization of Two-dimensional Airfoils Under Uncertain Conditions (Tech. Rep.). Hampton, Virginia: ICASE, NASA Langley Research Center.

Ilgin, H. E., & Gunel, M. H. (2007). The Role of Aerodynamic Modifications in the Form of Tall Buildings against Wind Excitations. *METU JFA*, 17–25.

Irwin, P. A. (2009). Wind Engineering Challenges of the New Generation of Super-Tall Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *97*(7-8), 328–334.

Jiang, Y., Alexander, D., Jenkins, H., Arthur, R., & Chen, Q. (2003). Natural Ventilation in Buildings: Measurement in a Wind Tunnel and Numerical Simulation with Large Eddy Simulation. *Journal of Wind Engineering and Industrial Aerodynamics*, *91*(3), 331–353.

Jianguang, Z. (2008). Interference Effects on Wind Loading of a Group of Tall Buildings in Close Proximity. Unpublished doctoral dissertation, The University of Hong Kong.

Jiao, X., & Heath, M. T. (2002). Feature Detection for Surface Meshes. In Proceedings of 8th International Conference on Numerical Grid Generation in Computational Field Simulations.

Jin, M., Zuo, W., & Chen, Q. (2012). Improvements of Fast Fluid Dynamics for Simulating Air Flow in Buildings. *Numerical Heat Transfer, Part B: Fundamentals: An International Journal of Computation and Methodology*, *62*(6), 419–438.

Jin, M., Zuo, W., & Chen, Q. (2013). Simulating Natural Ventilation in and Around Buildings by Fast Fluid Dynamics. *Numerical Heat Transfer, Part A: Applications*, *64*(4), 273–289.

Jin, Y. (2003). A Comprehensive Survey of Fitness Approximation in Evolutionary Computation. *Soft Computing*, *9*(1), 3–12.

Karagkouni, C. S., Fatah, A., Tsigkari, M., & Chronis, A. (2013). Façade Apertures Optimization: Integrating Cross-Ventilation Performance Analysis in Fluid Dynamics Simulation. In Proceedings of SimAUD SCS SpringSim'13.

Kareem, A., Spence, S., Bernardini, E., Bobby, S., & Wei, D. (2013). Using Computational Fluid Dynamics to Optimize Tall Building Design. *CTBUH Journal*, *2013*(III), 38–43.

Khanduri, A. C. (1997). Wind-Induced Interference Effects on Buildings - Integrating Experimental and Computerized Approaches. Unpublished doctoral dissertation, Concordia University, Montreal, CA.

Khanduri, A. C., Bedard, C., & Stathopoulos, T. (1997). Modelling Wind-Induced Interference Effects using Backpropagation Neural Networks. *Journal of Wind Engineering and Industrial Aerodynamics*, *72*, 71–79.

Khanduri, A. C., Stathopoulos, T., & Bedard, C. (1998). Wind-Induced Interference Effects on Buildings - A Review of the State-of-the-Art. *Engineering Structures*, *20*(7), 617–630.

Kolarevic, B., & Malkawi, A. M. (2005). Performative Architecture: Beyond Instrumentality (First ed.). New York and London: Spon Press, Taylor and Francis Group.

Krajnovic, S., & Davidson, L. (2002). Large-Eddy Simulation of the Flow Around a Bluff Body Introduction. *American Institute of Aeronautics and Astronautics*, *40*(5).

Krogstadt, P. A., & Antonia, R. A. (1999). Surface Roughness Effects in Turbulent Boundary Layers. *Experiments in Fluids*, *27*(5), 450–460.

Kwok, K. C. S., Wilkie, B. G., & Wilhelm, P. A. (1986). Effect of Edge Configuration on Wind-Induced Response of Tall Building. In 9th Australian Fluid Mechanics Conference, Auckland.

Lam, K. M., & To, A. P. (2006). Reliability of Numerical Computation of Pedestrian-level Wind Environment around a Row of Tall Buildings. *Wind and Structures*, *9*(6), 473–492.

Lam, K. M., Zhao, J. G., & Leung, M. Y. H. (2011). Wind-Induced Loading and Dynamic Responses of a Row of Tall Buildings under Strong Interference. *Journal of Wind Engineering and Industrial Aerodynamics*, *99*(5), 573–583.

Lamb, S., Kwok, K., & Walton, D. (2013). Occupant Comfort in Wind-Excited Tall Buildings: Motion Sickness, Compensatory Behaviours and Complaint. *Journal of Wind Engineering and Industrial Aerodynamics*, *119*, 1–12.

Laurence, D., & Mattei, J.-D. (1993). Current State of Computational Bluff-Body Aerodynamics. *Journal of Wind Engineering and Industrial Aerodynamics*, *49*, 23–43.

Lawrence, S., Giles, C. L., & Tsoi, A. C. (1996). What Size Neural Network Gives Optimal Generalization? Convergence Properties of Backpropagation (Tech. Rep.). Princeton; Australia; Maryland: NEC Research Institute, University of Queensland, University of Maryland.

Lawson, B. (2006). How Designers Think: The Design Process Demystified. Architectural Press.

Lee, B. E., & Fowler, G. R. (1975). The Mean Wind Forces Acting on a Pair of Square Prisms. *Building Science*, *10*, 107–110.

Lieberman, C., & Willcox, K. (2012). Goal-Oriented Inference: Approach, Linear Theory, and Application to Advection Diffusion. *SIAM Journal on Scientific Computing*, *34*(4), 1880–1904.

Lim, H. C., Thomas, T. G., & Castro, I. P. (2009). Flow Around a Cube in a Turbulent Boundary Layer: LES and Experiment. *Journal of Wind Engineering and Industrial Aerodynamics*, *97*(2), 96–109.

Liu, Y., Liu, X., & Wu, E. (2004). Real-Time 3D Fluid Simulation on GPU with Complex Obstacles. In 12th Pacific Conference on Computer Graphics and Applications, 2004 (pp. 247–256).

Lomax, H., Pulliam, T. H., & Zingg, D. W. (2001). Fundamentals of Computational Fluid Dynamics. Berlin: Springer.

Lu, S. C.-Y., Tcheng, D. K., & Yerramareddy, S. (1991). Integration of Simulation, Learning and Optimization to Support Engineering Design. *Annals of the CIRP*, *40*(1), 143–146.

Ly, H. V., & Tran, H. T. (1999). Modeling and Control of Physical Processes using Proper Orthogonal Decomposition. *Journal of Mathematical and Computer Modeling*, *33*(1), 223–236.

Malkawi, A. M. (2004). Developments in Environmental Performance Simulation. *Automation in Construction*, *13*(4), 437–445.

Malkawi, A. M., Srinivasan, R. S., Yi, Y. K., & Choudhary, R. (2005). Decision Support and Design Evolution: Integrating Genetic Algorithms, CFD and Visualization. *Automation in Construction*, *14*(1), 33–44.

Malkawi, A. M., Srinivasan, R. S., Yi, Y. K., Choudhary, R., & Arbor, A. (2003). Performance-Based Design Evolution: The Use of Genetic Algorithms and CFD. In Eighth International IBPSA Conference (pp. 793–798).

Marco, N., Stéphane, J.-a. D., & Lanteri, S. (1999). Multi-Objective Optimization in CFD by Genetic Algorithms (Tech. Rep.). Sophia Antipolis: Institut National de Recherche en Informatique et en Automatique.

Marsland, S. (2009). Machine Learning: An Algorithmic Perspective. Chapman & Hall/CRC, Taylor and Francis Group.

Maruta, E., Kanda, M., & Sato, J. (1998). Effects on Surface Roughness for Wind Pressure on Glass and Cladding of Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *74-76*, 651–663.

Matlab. (n.d.). TreeBagger algorithm. Retrieved from http://www.mathworks.co.uk/help/toolbox/stats/treebaggerclass.html

McNeel. (2013). Rhino. Retrieved 02.03.2014, from www.rhino3d.com

Meckesheimer, M., Booker, A. J., Barton, R. R., & Simpson, T. W. (2002). Computationally Inexpensive Metamodel Assessment Strategies. *AIAA Journal*, *40*(10).

Menicovich, D., Gallardo, D., Bevilaqua, R., & Vollen, J. O. (2002). Generation and Integration of an Aerodynamic Performance Database within the Concept Design Phase of Tall Buildings (Tech. Rep.). Troy, NY: Rensselaer Polytechnic Institute.

Merrick, R., & Bitsuamlak, G. (2009). Shape Effects on the Wind-Induced Response of High-Rise Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *6*(2), 1–18.

Moin, P., & Mahesh, K. (1998). Direct Numerical Simulation: A Tool in Turbulence Research. *Annual Review of Fluid Mechanics*, *30*(1), 539–578.

Montazeri, H., & Blocken, B. (2013). CFD Simulation of Wind-Induced Pressure Coefficients on Buildings with and without Balconies: Validation and Sensitivity Analysis. *Building and Environment*, *60*, 137–149.

Montazeri, H., & Blocken, B. (2014). Numerical Analysis of Surface Pressure Coefficients for a Building with Balconies. In 6th European and African Conference on Wind Engineering. Hamburg, Germany.

Mueller, V., Crawley, D. B., & Zhou, X. (2013). Prototype Implementation of a Loosely Coupled Design Performance Optimisation Framework. In Open Systems: Proceedings of the 18th International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2013) (pp. 675–684).

Mueller, V., & Strobbe, T. (2013). Cloud-Based Design Analysis and Optimization Framework. In Computation and Performance (eCAADe 2013) (Vol. 2, pp. 185–194).

Murakami, S. (1997). Current Status and Future Trends in Computational Wind Engineering. *Journal of Wind Engineering and Industrial Aerodynamics*, *67-68*, 3–34.

Murakami, S., Mochida, A., Hayashi, Y., & Sakamoto, S. (1992). Numerical Study on Velocity-Pressure Field and Wind Forces for Bluff Bodies by k-e , ASM and LES. *Journal of Wind Engineering and Industrial Aerodynamics*, *44*, 2841–2852.

Obayashi, S., & Sasaki, D. (2003). Visualization and Data Mining of Pareto Solutions using Self-Organizing Map. In Evolutionary multi-criterion optimization (pp. 796–809). Springer Berlin Heidelberg.

Ochoa, J. S., & Fueyo, N. (2004). Large Eddy Simulation of the Flow past a Square Cylinder. *PHOENICS Journal of Computational Fluid Dynamics and its Applications*, *2004*(17).

Oldfield, P., Trabucco, D., & Wood, A. (2014). Roadmap on the Future Research Needs of Tall Buildings (Tech. Rep.). Chicago: Council on Tall Buildings and Urban Habitat.

Ong, Y. S., Nair, P. B., & Keane, A. J. (2003). Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling. *AIAA Journal*, *41*(4), 687–696. Retrieved from http://arc.aiaa.org/doi/abs/10.2514/2.1999

Panofsky, H. A., & Dutton, J. A. (1984). Atmospheric Turbulence. Wiley.

Park, S. M., Elnimeiri, M., Sharpe, D. C., & Krawczyk, R. J. (2004). Tall Building Form Generation by Parametric Design Process. In CTBUH (pp. 1–7). Seoul Conference.

Peters, B., & Peters, T. (2013). Inside SmartGeometry (First ed.). John Wiley and Sons.

Popper, K. (1959). The Logic of Scientific Discovery. Hutchinson & Co.

Post, F. H., Vrolijk, B., Hauser, H., Laramee, R. S., & Doleisch, H. (2002). Feature Extraction and Visualisation of Flow Fields. In Eurographics.

Ramanathan, S., & Graening, L. (2009). Knowledge Incorporation into Evolutionary Algorithms to speed up Aerodynamic Design Optimizations. Unpublished doctoral dissertation, Universitat Stuttgart.

Rendall, T. C. S., & Allen, C. B. (2008). Multi-Dimensional Aircraft Surface Pressure Interpolation using Radial Basis Functions. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, *222*(4), 483–495.

RIBA. (2013). RIBA Plan of Work. Retrieved 15.01.2014, from www.ribaplanofwork.com

Richards, P. J., Hoxey, R. P., Connell, B. D., & Lander, D. P. (2007). Wind-Tunnel Modelling of the Silsoe Cube. *Journal of Wind Engineering and Industrial Aerodynamics*, *95*(9-11), 1384–1399.

Richards, P. J., Hoxey, R. P., & Short, L. J. (2001). Wind Pressures on a 6m Cube. *Journal of Wind Engineering and Industrial Aerodynamics*, *89*(14-15), 1553–1564.

Rittel, H. W. J., & Webber, M. M. (1973). Dilemmas in a General Theory of Planning. *Policy Sciences*, *4*, 155–169.

Roache, P. J. (1997). Quantification of Uncertainty in Computational Fluid Dynamics. *Annual Review of Fluid Mechanics*, *29*(1), 123–160.

Robinson, T. D., Eldred, M. S., Willcox, K., & Haimes, R. (2008). Surrogate-Based Optimization using Multifidelity Models with Variable Parameterization and Corrected Space Mapping. *AIAA Journal*, *46*(11), 2814–2822.

Rodi, W. (1993). On the Simulation of Turbulent Flow past Bluff Bodies. *Journal of Wind Engineering and Industrial Aerodynamics*, *46-47*, 3–19.

Rodi, W. (1997). Comparison of LES and RANS Calculations of the Flow around Bluff Bodies. *Journal of Wind Engineering and Industrial Aerodynamics*, *69-71*, 55–75.

Rofail, A. W., & Kwok, K. C. S. (1999). The Effect of Sunshading Elements on Cladding Pressures. In 10th International Conference on Wind Engineering.

Rohrlich, F. (1990). Computer Simulation in the Physical Sciences. *Philosophy of Science Association*, *2*, 507–518.

Sakamoto, H., & Haniu, H. (1988). Aerodynamic Forces Acting on Two Square Prisms placed Vertically in a Turbulent Boundary Layer. *Journal of Wind Engineering and Industrial Aerodynamics*, *31*, 41–66.

Samarasinghe, S. (2007). Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition. Auerbach Publications, New York.

Samareh, J. A. (2001). Survey of Shape Parameterization Techniques for High-Fidelity Multidisciplinary Shape Optimization. *American Institute of Aeronautics and Astronautics*, *39*(5), 333–343.

Saunders, J. W., & Melbourne, W. H. (1979). Buffeting Effects of Upwind Buildings. In Proceedings of the 5th International Conference on Wind Engineering (pp. 593–605). Fort Collins, CO.

Schilders, W. (2008). Introduction to Model Order Reduction. In Model Order Reduction: Theory, Research Aspects and Applications (13th ed.). Springer.

Selvam, R. P. (2008). Developments in Computational Wind Engineering. *Journal of Wind Engineering and Industrial Aerodynamics*, *5*(1), 47–54.

Shah, K. B., & Ferziger, J. H. (1997). A Fluid Mechanician's View of Wind Engineering: Large Eddy Simulation of Flow past a Cubic Obstacle. *Journal of Wind Engineering and Industrial Aerodynamics*, *67-68*, 211–224.

Shahrokhi, A., & Jahangirian, A. (2007). An Efficient Aerodynamic Optimization Method using a Genetic Algorithm and a Surrogate Model. In 16th Australian Fluid Mechanics Conference (pp. 475–480).

Sheikholeslami, M. (2009). You Get More than You Make. Unpublished doctoral dissertation, Simon Fraser University.

Shneiderman, B. (2007). Accelerating Discovery and Innovation. *Communications of the ACM*, *50*(12).

Simon, H. (1996). The Science of the Artificial (3rd ed.). MIT Press.

Simpson, T. W., Mauery, T. M., Korte, J. J., & Mistree, F. (2001). Kriging Models for Global Approximation in Simulation-Based Multidisciplinary Design Optimization. *AIAA Journal*, *39*(12), 2233–2241.

Simpson, T. W., Peplinski, J. D., Koch, P. N., & Allen, J. K. (2001). Metamodels for Computer-based Engineering Design: Survey and Recommendations. *Engineering with Computers*, *17*, 129–150.

SISTA. (2012). LS-SVM Publications. Retrieved 19.08.2012, from http://www.esat.kuleuven.be/sista/lssvmlab/

Som, S., Senecal, P. K., & Pomraning, E. (2012). Comparison of RANS and LES Turbulence Models against Constant Volume Diesel Experiments. In ILASS Americas, 24th Annual Conference on Liquid Atomization and Spray Systems (Vol. Di). San Antonio, TX.

Srinivasan, R. S., & Malkawi, A. M. (2004). The Use of Learning Algorithms for Real-Time Immersive Data Visualisation in Buildings. In SIGRADI (pp. 329–332).

Stam, J. (1999). Stable Fluids (Tech. Rep.). Seattle: Alias-Wavefront.

Stathopoulos, T. (1997). Computational Wind Engineering: Past Achievements and Future Challenges. *Journal of Wind Engineering and Industrial Aerodynamics*, *67-68*, 509–532.

Stathopoulos, T., & Zhu, X. (1988). Wind Pressures on Buildings with Appurtenances. *Journal of Wind Engineering and Industrial Aerodynamics*, *31*, 265–281.

Stathopoulos, T., & Zhu, X. (1990). Wind Pressure on Buildings with Mullions. *Journal of Structural Engineering*, *116*(8), 2272–2291.

Suykens, J. A. K. (2008). Data Visualization and Dimensionality Reduction using Kernel Maps with a Reference Point. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, *19*(9), 1501–17.

Suykens, J. A. K., Gestel, T. V., Brabanter, J. D., Moor, B. D., & Vandewalle, J. (2011). LS-SVM Toolkit. Retrieved 17.08.2012, from http://www.esat.kuleuven.be/sista/lssvmlab/

Suykens, J. A. K., & Vandewalle, J. (1999). Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, *9*, 293–300.

Szilvasi-Nagy, M. (2006). About Curvatures on Triangle Meshes. In KoG 10 - Journal of Croatian Society for Geometry and Graphics (pp. 13–18).

Tamura, T., Nozawa, K., & Kondo, K. (2008). AIJ Guide for Numerical Prediction of Wind Loads on Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *96*(10-11), 1974–1984.

Tamura, Y. (2009). Wind and Tall Buildings. In EACWE 5 (Vol. 1). Florence, Italy.

Tamura, Y., Tanaka, H., Ohtake, K., Nakai, M., & Kim, Y. (2010). Aerodynamic Characteristics of Tall Building Models with Various Unconventional Configurations. In Structures Congress (pp. 3104–3113). ASCE.

Tanaka, H., Tamura, Y., Ohtake, K., Nakai, M., & Kim, Y. (2012). Experimental Investigation of Aerodynamic Forces and Wind Pressures acting on Tall Buildings with Various Unconventional Configurations. *Journal of Wind Engineering and Industrial Aerodynamics*, *107-108*(2012), 179–191.

Taniike, Y. (1992). Interference Mechanism for Enhanced Wind Forces on Neighboring Tall Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *42*, 1073–1083.

Taniike, Y., & Inaoka, H. (1988). Aeroelastic Behavior of Tall Buildings in Wakes. *Journal of Wind Engineering and Industrial Aerodynamics*, *28*(1-3), 317–327.

Tcheng, D. K., Lambert, B., Lu, S. C.-Y., & Rendell, L. (1989). Building Robust Learning Systems by Combining Induction and Optimization. In Proceedings of the 11th International Joint Conference on Artificial Intelligence (pp. 806–812). San Mateo, CA.

Tominaga, Y., Mochida, A., Yoshie, R., Kataoka, H., Nozu, T., Yoshikawa, M., & Shirasawa, T. (2008). AIJ Guidelines for Practical Applications of CFD to Pedestrian Wind Environment around Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *96*(10-11), 1749–1761.

Tse, K. T., Hitchcock, P. A., Kwok, K. C. S., Thepmongkorn, S., & Chan, C. M. (2009). Economic Perspectives of Aerodynamic Treatments of Square Tall Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *97*(9-10), 455–467.

Turk, G., & Levoy, M. (1994). The Stanford 3D Scanning Repository. Retrieved 06.03.2013, from http://graphics.stanford.edu/data/3Dscanrep/

Turner, C. J., Crawford, R. H., & Campbell, M. I. (2007). Global Optimization of NURBs-based Metamodels. *Engineering Optimization*, *39*(3), 245–269.

Vapnik, V. N. (2000). The Nature of Statistical Learning Theory (2nd ed.). Springer.

Vogl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., & Alkon, D. L. (1988). Accelerating the Convergence of the Back-propagation Method. *Biological Cybernetics*, *59*(4-5), 257–263.

Walter, N., Laligant, O., & Aubreton, O. (2008). Salient Point SUSAN 3D Operator for Triangles Meshes. In The 2nd International Topical Meeting on Optical Sensing and Artificial Vision.

Wang, G. G., & Shan, S. (2007). Review of Metamodeling Techniques in Support of Engineering Design Optimization. *Journal of Mechanical Design*, *129*(4), 370.

Watson, B., Mueller, P., Wonka, P., Sexton, C., Veryovka, O., & Fuller, A. (2008). Procedural Urban Modeling in Practice (Tech. Rep. No. June). IEEE: Computer Graphics and Applications.

Watts, S., Kalita, N., & Maclean, M. (2007). The Economics of Super-Tall Towers. *The Structural Design of Tall and Special Buildings*, *16*, 457–470.

Weinkauf, T., Hege, H.-C., & Theisel, H. (2012). Advected Tangent Curves: A General Scheme for Characteristic Curves of Flow Fields. *Computer Graphics Forum*, *31*(2pt4), 825–834.

Wicke, M., Stanton, M., & Treuille, A. (2009). Modular Bases for Fluid Dynamics. *ACM Transactions on Graphics*, *28*(3).

Wilkinson, S. (2011). Transient Wind-Load Optimisation using Fast Fluid Dynamics and Multi-Conditional Genetic Algorithms. Master's thesis, University College London.

Willcox, K. (2000). Reduced-Order Aerodynamic Models for Aeroelastic Control of Turbomachines. Unpublished doctoral dissertation, MIT.

Wilson, A. (2010). Knowledge Power: Interdisciplinary Education for a Complex World. Routledge.

Winsberg, E. (2008). Sanctioning Models: The Epistemology of Simulation. *Science in Context*, *12*(02).

Woodbury, R. (2010). Elements of Parametric Design (First ed.). New York, NY: Routledge.

Wright, N. G., & Easom, G. (2003). Non-linear kE Turbulence Model Results for Flow over a Building at Full-scale. *Applied Mathematical Modelling*, *27*(12), 1013–1033.

Xie, J. (2014). Aerodynamic Optimization of Super-Tall Buildings and its Effectiveness Assessment. *Journal of Wind Engineering and Industrial Aerodynamics*, *130*(2014), 88–98.

Xie, Z.-N., & Gu, M. (2004). Mean Interference Effects among Tall Buildings. *Engineering Structures*, *26*(9), 1173–1183.

Yerramareddy, S., Tcheng, D. K., Lu, S. C.-Y., & Assanis, D. N. (1992). Creating and Using Models for Engineering Design: A Machine Learning Approach. *IEEE Expert*, *7*(3), 52–59.

Zhang, A., Gao, C., & Zhang, L. (2005). Numerical Simulation of the Wind Field around Different Building Arrangements. *Journal of Wind Engineering and Industrial Aerodynamics*, *93*(12), 891–904.

Zhang, A., & Gu, M. (2008). Wind-Tunnel Tests and Numerical Simulations of Wind Pressures on Buildings in Staggered Arrangement. *Journal of Wind Engineering and Industrial Aerodynamics*, *96*(10-11), 2067–2079.

Zhang, A., & Zhang, L. (2004). RBF Neural Networks for the Prediction of Building Interference Effects. *Computers & Structures*, *82*(27), 2333–2339.

Zhang, D.-y., Luo, Y.-h., Li, X., & Chen, H.-w. (2011). Numerical Simulation and Experimental Study of Drag-Reducing Surface of a Real Shark Skin. *Journal of Hydrodynamics, Ser. B*, *23*(2), 204–211.

Zingg, D. W., & Elias, S. (2006). Aerodynamic Optimization under a Range of Operating Conditions. *AIAA Journal*, *44*(11), 2787–2792.

Zuo, W., & Chen, Q. (2009). Real-Time or Faster-than-Real-Time Simulation of Airflow in Buildings. *Indoor Air*, *19*(1), 33–44.

Zuo, W., & Chen, Q. (2010). Fast and Informative Flow Simulations in a Building by using Fast Fluid Dynamics Model on Graphics Processing Unit. *Building and Environment*, *45*(3), 747–757.

Appendix A

Publications

A.1 Wilkinson, S., Bradbury, G., and Hanna, S. (2015). Reduced-Order Urban Wind Interference. *Journal of Simulation*.

Reduced-Order Urban Wind Interference

Abstract

A novel approach is demonstrated to approximate the effects of complex urban interference on the wind-induced surface pressure of tall buildings. This is achieved by decomposition of the domain into two components: the obstruction model (OM) of the static large-scale urban context, for which a single computational fluid dynamics (CFD) simulation is run; and the principal model (PM) of the isolated tall building under design, for which repeatable reduced-order model (ROM) predictions can be made. The ROM is generated with an artificial neural network (ANN), using a set of feature vectors comprising an input of local shape descriptors and a range of wind speeds from a training geometry, and an output response of pressure. For testing, the OM CFD simulation provides the flow boundary condition wind speeds to the PM ROM prediction. The result is vertex-resolution surface pressure data for the PM mesh, intended for use within generative design exploration and optimisation. It is found that the mean absolute prediction error is around 5.0% (σ:7.8%) with an on-line process time of 390s, 27-times faster than conventional CFD simulation; considering full process time, only 3.2 design iterations are required for the ROM time to match CFD. Existing work in the literature focuses solely on creating generalised rules relating global configuration parameters and a global interference factor (IF). The work presented here is therefore a significantly alternative approach, with the advantages of increased geometric flexibility, output resolution, speed, and accuracy.

1 Introduction

In the context of wind simulation for tall buildings, meaningful results can only be achieved through an appropriate application of boundary conditions. One example is the effect of the surrounding environment on the wind field around a chosen building. In practice, the difference between flow behaviour with and without such context, termed 'wind interference,' can have a significant effect on predictions.

Computational fluid dynamic (CFD) analysis in architectural design typically involves response times that are obstructive to the fast iterations of contemporary generative practice. In this parametric paradigm, architects can easily generate immense numbers of alternative scenarios but are then faced with the time-consuming task of evaluation and selection. The assessment of isolated tall buildings in itself is an intensive task, which is exacerbated when extending the scope to include context. A coarser resolution is required due to the larger domain size and computational restrictions, therefore slower simulations means fewer options can be evaluated and optimisation is infeasible.

A previous proposal by Wilkinson et al. (2013) demonstrated the speed and accuracy of a reduced-order model (ROM) based on the use of a geometric feature vector ($P\{z,\mathbf{n},\mathbf{n}\sigma^r,\mathbf{u}\}$). This was applied to the prediction of wind-induced surface pressure on isolated tall buildings, and aimed at parametric CAD tools such as *GenerativeComponents*. The ROM was generated with an artificial neural network (ANN) trained on a set of procedural tall building models which are evaluated with steady, time-averaged RANS (Reynolds-Averaged Navier-Stokes) CFD. A limitation of this previous work however was the exclusion of surrounding context; that is, the predictions were in isolation with unrealistic boundary conditions.

In our work, their problem definition and feature vector is extended to include local fluid properties ($P\{v,\mathbf{n},\mathbf{n}\sigma^r,\mathbf{u}\}$) in order to support complex urban scenarios. For the training set, v_S is derived as the vertex's upstream wind speed from a set of principal models (PMs, the isolated design geometry) evaluated under a range of wind speeds; whereas for the test set, v_T is derived from the vertex's position in the field of the obstruction model (OM, the context geometry without the design). This equates to a superimposition or combination of a large-scale, one-off contextual CFD simulation and a small-scale, repeatable design ROM prediction.

Many attempts have been made to approximate or generalise wind interference, i.e. the effect of multiple buildings in the domain (Tables 1 and 2). However, all have relied on a top-down problem definition in relating the position of identical surrounding building cuboids with a global Interference Factor (IF) for the design building. Significant improvement can be made here by: (i) increasing the geometric complexity of both the context and design models; and (ii) increasing output resolution from a single global factor to the vertex-level.

In this paper, a background review is initially presented to identify existing limitations, followed by the proposed methodology. This is then demonstrated for a realistically complex design case along with parametric sensitivity analyses on the training set size, number of required training simulations, and location of the test wind speed; leading to a discussion on the speed, accuracy, and limitations of the method.

1.1 Contribution

The primary aim of this work is to test the scalability of the ROM to cases with complex urban interference by extending the shape-based feature vector to include local wind speed. This work is therefore a development on the methodology and results of Wilkinson et al. (2014, 2013). Predictions of the isolated tall buildings are inadequate when considering the significant effects that dense urban environments can have on the wind-induced surface pressure. Following this, existing methods found in the literature to include interference are both limiting and not amenable to the basic ROM methodology (§3.2). The difference in approaches is fundamental: existing work focuses on explicitly describing the OM, PM, and their relation to one another with global parameters; our approach describes the PM through local shape features and the effect of the OM implicitly through local wind speeds.

2 Literature Review

Our work investigates the design of tall buildings, particularly with respect to the use of generative CAD tools. In conjunction with these tools, computational analysis can be used for guidance, exploration and optimisation of an increasingly broad selection of potential designs. The review covers analysis for generative design and the time-accuracy tradeoffs inherent involved, approaches to resolving this problem through solver (CFD) and solution (model reduction) approximation, and finally sensitivity analyses and generalisation of wind interference.

2.1 Performance-Based Generative Design

In current generative design practice, enabled by the ubiquity of computation and advances in computer aided design, integrating performance behaviours into generative models has entered the foreground (Malkawi, 2004). Examples can be seen in the use of structural, energy and thermal, materiality, fabrication, and air movement (either internally for comfort and indoor air quality; or externally for structural or facade aerodynamics, pedestrian comfort, or pollution dispersal).

Air movement, predicted through CFD, suffers the most from restrictive response times, predominantly because of the historical focus on accuracy rather than speed (due to low-tolerance high-risk scenarios in aerospace engineering). Arguably, the margins for acceptable error are more tolerant in building design, meaning that the simulation accuracy requirements can be relaxed or traded off for speed improvements (particularly at early design stages).

In these early stages of light-weight (fast and less-accurate) performance feedback, there can be more allowance for design exploration and optimisation. This is supported by the idea of speed-accuracy trade-offs (SATs) (Chittka et al., 2009), which suggests that for low-risk problems, it is often better to make faster, less accurate decisions. In other words, in the scope of the larger problem of building design, it is better to have a broader perspective on the performance variability rather than an extremely accurate but narrow perspective on fewer cases.

2.2 Time-Accuracy Tradeoffs

CFD simulation can be one of the most intensive and time-consuming stages in the assessment of building performance. Difficulty therefore typically arises in guidance for early project stage decisions due to the slow feedback from conventional CFD approaches. Paradoxically, although the simulations are most valuable at early stages, the slow-and-accurate CFD simulation is better suited to later stages where the design scope is more constrained. This leads to the fundamentally opposing requirements of a fast-yet-accurate tool.

It is therefore prudent to consider compromises in the speed-accuracy trade-off during these early stages; that is, by sacrificing accuracy for speed, so that more design options can be explored. This ongoing problem was recognised over two decades ago by Lu et al. (1991), as the need for both speed and simulation accuracy to meets the demands of early design stages. They generate a range of reduced-order models of a combustion engine simulation, with varying accuracy and speed that can be used throughout the design process. Their solution is posed as a Pareto front of non-dominated solutions, rather than a simpler trade-off curve based on biological decision making as suggested by Chittka et al. (2009).

Approximation of some description is key to this relationship between time and accuracy. An ideal approximation of reality can replicate it instantly with no error, at least for the variables of interest. In our work, the variable of interest is wind-induced surface pressure. This in itself is a product of a number of field variables which can effectively be discarded, so long as the pressure is approximated well. This is the fundamental premise of modelling, of which approximation is a like-term, and of which two distinct approaches will now be introduced.

2.3 Solver Approximation

Most approaches towards CFD approximation focus on simplification of the solver itself. For instance: simplified meshes (spatial discretisation); the use of lower-order equations; or the treatment of turbulence through modelling. These methods can be classed as type-one, *solver approximation* (Figure 1). For instance, RANS (Reynolds-Averaged Navier-Stokes), LES (Large Eddy Simulation), and DNS (Direct Numerical Simulation) all treat turbulence with different numerical approaches, i.e. temporally, spatially, and directly.

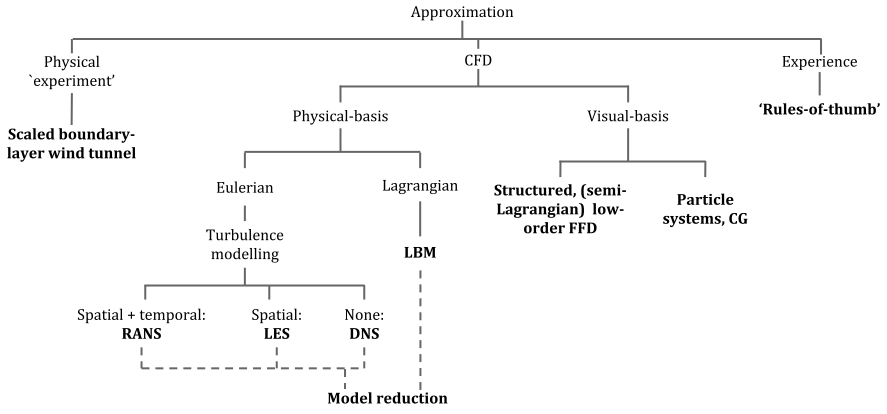


Figure 1: CFD solver approximation taxonomy.

Another example is the 'Stable Fluids' fast fluid dynamics (FFD) solver developed by Stam (1999) for the computer graphics and games industries, which has subsequently been developed and tested for architectural applications (Athanailidi et al., 2014; Chronis et al., 2012, 2011; Karagkouni et al., 2013). Development and application for architectural design was motivated by three factors: a limited, low Reynolds number validation which suggested it as suitable for purposes beyond the scope of the validation (Zuo & Chen, 2009, 2010); the qualitative appearance of accuracy for turbulent flows; and its remarkable speed compared to traditional CFD methods like RANS (Reynolds-Averaged Navier-Stokes). Zuo & Chen (2009) implemented the FFD with a zero-equation turbulence model but found that it performed worse since it was not designed or

suited to the FFD approach. It should be noted, however, that with a lack of turbulence model, the solver relies on continuous interaction (such as game character movement) to compensate for numerical dissipation.

The benefit of solver over solution approximation is the availability of full fluid field data, although production of surface data is more difficult due to the structured mesh approximation (voxelisation).

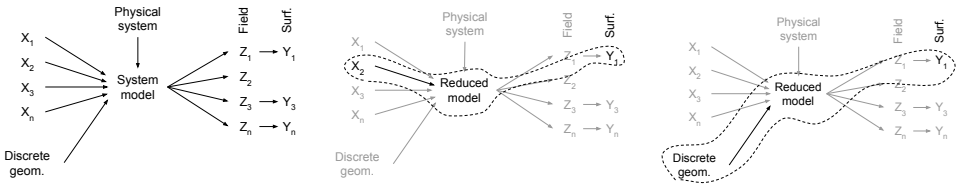
2.4 Solution Approximation

Another possible approach to this problem, type-two, is *solution approximation*. CFD originated in aeronautics and astronautics, as such there is a large quantity of work directed towards modelling and optimisation of aerofoils, fuselages, and turbine blades. An optimisation routine will often generate large data sets of simulation data, from which knowledge of the problem can be extracted. The following fall into the greater category of supervised machine learning approaches where the relationship between an input feature vector extracted from some geometry and the ground truth data output from full CFD simulation is learnt.

In one such case, a large model set of turbine blades is used with a decision tree to analyse the relationship between point deformation of models and their change in surface pressure (Graening et al., 2008; Graening & Sendhoff, 2014). Areas of high sensitivity can then be mapped onto a pre-defined base geometry and used to focus subsequent analysis. Ramanathan & Graening (2009) extend this work further to incorporate an evolutionary optimisation process, so as to use the information extracted from previous cases to create non-random initial populations of solutions and to guide the evolution.

Analyses that are potentially obstructive to the design process may involve partial differential equations (PDEs), such as the Navier-Stokes equations of fluid flow and the Maxwell equations in electromagnetism (Degroote et al., 2010). Whilst these methods give high-accuracy results, they are computationally expensive and cannot be computed in real-time. As a result, a design process using high-accuracy techniques has inherently slow response times and loses any desired interactivity.

Significant efforts have been made to reduce the complexity of these systems in order to make them interactive; this is generally referred to as *model reduction*. Reduced-order models (ROMs) approximate representations of system behaviours, namely for computational simulations with slow response times; with the aim to create a lower-dimensional system model whilst retaining predictive fidelity (Bui-Thanh et al., 2008; Schilders, 2008).



(a) Simulation (b) Typical ROM (c) Proposed ROM

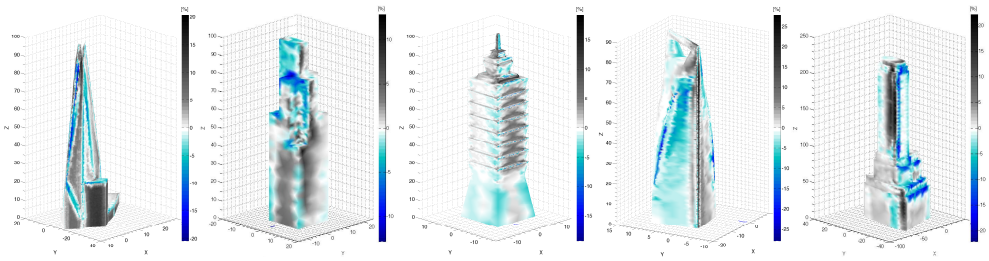
Figure 2: Reduced-order model schematic.

Rendall & Allen (2008) use spatial and behavioural parameters as input feature vectors to a radial basis function (RBF). The RBF is used to interpolate and merge CFD and wind-tunnel data on pressure coefficient values (lift and drag) for aerofoil analysis. They use an input feature vector $C_p\{x, y, z, a, M, Re\}$: where x, y, z is the spatial position; a the angle of attack; M the Mach number; and Re the Reynolds number.

Whilst this method proved successful for linking behavioural characteristics (a, M, and Re) to data sources (CFD and wind-tunnel), it is limited to a single geometry, thus the use of explicit spatial positions (x, y, and z). For cases of differing geometry between training and testing, spatial positions become non-unique and can therefore not be used within the feature vector. This necessitates the use of either explicit global design parameters or implicit local shape description.

Using spatial positions (or mesh node numbers) for a feature vector is also proposed by Srinivasan & Malkawi (2004). In this case, an ANN is used to predict post-processed CFD data for rapid visualisation and interpolation of boundary conditions with an augmented reality user display system. The input feature vector, $\mathbf{X}$, and output response, $\mathbf{Y}$, are defined as: $\mathbf{X}\{n, S, P\}, \mathbf{Y}\{T, V\}$, where T is the air temperature and V is the air speed at a node, n is the mesh node number (1224 nodes in the cubic room), S is the supply temperature, and P is the supply pressure. Again the proposal is strongly limited to not only a single geometry but to a single mesh by the use of spatial positions or node numbers corresponding consistently with fixed locations. The limitation again is that for differing training/test geometry the positions are non-unique.

Wilkinson et al. (2013) developed the ROM feature vector using $P\{z,\mathbf{n},\mathbf{n}\sigma^r,\mathbf{u}\}$ for tall building surface pressure prediction. Predictions are made through training an ANN on these local shape features extracted from a set of procedural tall building models evaluated with RANS. The separate test set of 10 models is a selection of existing tall building models (Figure 3), showing results of a mean absolute error of between 1.994 and 4.440% (σ:2.096 to 5.088%), and an on-line response time of between 14.720 and 809.983s.



(a) (b) (c) (d) (e)

Figure 3: ROM test model prediction errors, Y'-Y [%]: (a) The Shard; (b) Willis Tower; (c) Taipei 101; (d) Shanghai World Financial Centre; (e) Exchange Place. (Wilkinson et al., 2013)

Existing work on extracting features from the fluid field is primarily focused on knowledge extraction or data mining from large sets. Post et al. (2002) review a number of applications such as flow topology classification, vortex identification and tracking, shock wave detection, and separation/attachment detection. Identification of flow characteristics (vortex cores, separation/attachment, and shock waves) can be 'mined' during a CFD simulation (Gosnell et al., 2012). In extremely high resolution problems (up to 300 million grid-point transonic turbofan simulation with RANS) it can often take weeks or months for a single run to converge on a stable result. Therefore employing feature detection during the simulation can give insight into the development and stability of the actual fluid structure rather than physical properties of the simulation (e.g. mass and momentum residuals). This results in considerable time-savings if features of interest can be observed and tracked directly, potentially allowing the user to cut short the simulation when they are confident that the flow is stable enough for the accuracy requirements of the problem.

2.5 Wind Interference

Interference refers to the positive or negative effect that nearby buildings may have upon the wind behaviour of one another. Within an urban situation this is very common, and since the effects can be significant it is necessary to consider the context within the simulation; that is, independently designed buildings can not be treated in isolation.

A common misconception is that interference always reduces wind loads in comparison to the isolated case. Whilst this may be true for a uniformed array of similar buildings in close proximity, wind loads can be increased in the more complex, realistic case. The key factors in determining the effects of interference are the size, shape, and configuration of the buildings with respect to the direction of flow. The effects have been shown to be as great as up to 46% under-prediction and 525% over-prediction from regulatory loads on simple prismatic buildings

(Stathopoulos, 1984). An over-prediction of wind pressure is less dangerous than an under-prediction, since the latter may cause discomfort or safety issues. Khanduri et al. (1998) present a thorough review of the full past and present state of research in interference. A summary of typical studies can be found in Table 1.

Within the paradigm of such global parametric analysis, simplifications of both the problem and the solution are necessary. For all the cases in Table 1, simple cuboids are used with typical variables such as aspect ratio and position configuration; in other words, translating the objects over the two-dimensional horizontal plane. These studies analysed the effects of a small number of adjacent structures, leading to the development of the Interference Factor (IF). This is a ratio between the wind loads with and without the interference from adjacent structures. No such studies, however, have been undertaken which consider realistically complex shapes or contexts since they are typically esoteric and difficult to generalise.

Table 1: Summary of existing interference global parameter sensitivity studies.

No.	**Evaluation method**	**O.**	**SD.**	**AR.**	**C.**	α	**Source**
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Lee & Fowler (1975)
2	WT		$\bullet_{X,Y}$	$\bullet$		-	Sakamoto & Haniu (1988)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Taniike (1992)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Saunders & Melbourne (1979)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Bailey & Kwok (1985)
2	WT		$\bullet_{X,Y}$	$\bullet$		0.14	Taniike & Inaoka (1988)
2	WT	$\bullet$	$\bullet_{X,Y}$	$\bullet_{Z}$		0.19	Agrawal et al. (2012)
2	WT + CFD (RNG k-ϵ)	$\bullet$				0.16	Zhang & Gu (2008)
2	WT		$\bullet_{X,Y}$	$\bullet_{X,Y}$		0.14	Taniike & Inaoka (1988)
2 & 3	WT		$\bullet_{X,Y}$	$\bullet_{X,Z}$		0.16	Xie & Gu (2004), Gu & Xie (2011)
5	WT	$\bullet$	$\bullet$		$\bullet$	$\bullet$	Lam et al. (2011)
5	WT	$\bullet$		$\bullet_{X,Y}$		0.15	Jianguang (2008)
Multi.	CFD (RNG k-ϵ)	$\bullet$			$\bullet$	0.22	Zhang et al. (2005)

$\bullet$ varied in study; - no data; WT wind-tunnel; **No.** Number of Study Buildings; **O.** Orientation; **SD.** Separation Distance; **AR.** Aspect Ratio; **C.** Configuration; α Wind profile exponent; X is direction perpendicular to flow; Y stream-wise; and Z vertical.

2.6 Interference Approximation

In a few cases generalisation, or regression, has been attempted (Table 2) with the IF used as output response and basic scenario parameters as input features.

Table 2: Summary of existing interference global parameter generalisation studies.

No.	**Evaluation / Regression method**	**O.**	**SD.**	**AR.**	**C.**	α	**Source**
2	WT / Polynomial		$\bullet_{X,Y}$	$\bullet$		0.14	Khanduri (1997)
2	WT / RBF		$\bullet_{X}$	$\bullet$		$\bullet$	English & Fricke (1999)
2	WT / RBF		$\bullet_{X,Y}$				Khanduri et al. (1997)
2	WT / RBF	$\bullet$	$\bullet_{X,Y}$	$\bullet$		$\bullet$	Khanduri (1997)
2	WT / RBF		$\bullet$	$\bullet$		$\bullet$	Zhang & Zhang (2004)

No. Number of Study Buildings; **O.** Orientation; **SD.** Separation Distance; **AR.** Aspect Ratio; **C.** Configuration; α Wind profile exponent; X is direction perpendicular to flow; Y stream-wise; and Z vertical.

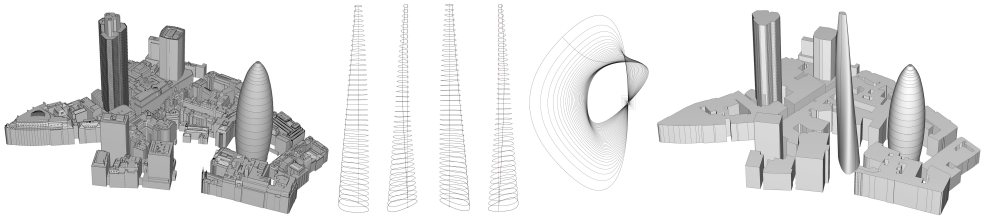
In the first case, regression curves are fitted with either second-, third-, or fourth-order polynomials; whilst in the remaining majority a radial basis function (RBF) ANN was used. In both cases, the IF was collected from new or existing wind-tunnel data. The limitations of this approach are the simplistic geometries (cuboids of a single height), basic configurations (typically two or three buildings), and lack of output data (only a single performance metric: the IF, rather than the varied surface pressure distribution). It should be noted that in every case the studies were constrained to a limited number of cuboids, a significant simplification in attempting to create generalised interference rules.

3 Methodology

The approach here is to combine: i) a large-scale CFD simulation of an urban context, the *obstruction model (OM)*, which remains static throughout the design process and can therefore be simulated only once; and ii) a small-scale ROM prediction of an isolated tall building, the *principal model (PM)*, which can be iteratively changed through generative design.

Considerable time can be saved if it can be demonstrated that a CFD simulation can effectively be used for boundary conditions to a ROM. The clear advantage is that the full OM does not need to be re-run with every change of PM.

A realistic OM of the dense City district in London is used (Figure 4a), along with a realistic design PM (Figure 4b). These are put together for the full validation model, OM+PM (Figure 4c). Note that the geometry in Figure 4a also shows the level of detail typically found from source, and Figure 4c shows the same geometry after simplification and meshing. The geometry intends to replicate a scenario that would be found in practice. The design model (Figure 4b) is arbitrary, but is based on prior models generated at competition, massing, or form-finding project stages within practice. The design model has a height of $310m$ (Z-axis), a cross-wind (X-axis) width of $55.4m$, and an along-wind (Y-axis) depth of $41.8m$; the aspect ratio (width:height) is therefore roughly 1:6. In comparison, the upstream Swiss Re is $180m$ and the downstream Tower 42 is $183m$.



(a) OM (b) PM (c) OM+PM

Figure 4: Components of the principal model (PM) and obstruction model (OM).

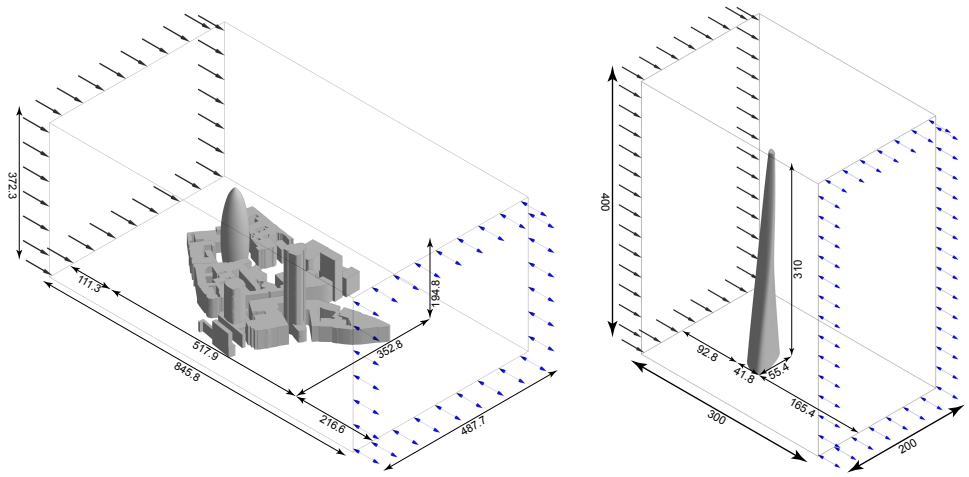
3.1 Simulation Methodology

CFX 13.0 (ANSYS, 2014) is used for the steady-state Reynolds-Averaged Navier-Stokes (RANS) simulations with a k-ϵ turbulence model. Typically the models are meshed with roughly an equal number of cells (up to the maximum available computational resources), of around four million elements. The PM simulations, for the training set, therefore have a finer resolution than the OM. The geometry is created in *GenerativeComponents* (Bentley Systems, 2013). Process times are based on a 2.66GHz *i*7 quadcore, 4GB RAM.

The CFD simulation domains and significant dimensions are given for the OM and PM in Figure 5. For the ground, a no slip smooth wall is assigned (i.e. fluid velocity at wall boundary is zero); for the sides and top parallel to the flow, a free slip wall (i.e. zero shear stress from wall friction); and for the outlet, a zero relative pressure opening. For the inlet, the wind profile is applied as described below, with a medium intensity turbulence and eddy viscosity ratio (ANSYS, 2009).

Basic simulation parameters are: high-resolution advection and turbulence numerics; isothermal fluid at $25°C$; a scalable wall function; and a convergence residual target of $1.0e^{-6}$ RMS. The following meshing parameters are used: an unstructured tetrahedral domain mesh, with patch independence; a boundary surface element size of $5m$; a model surface minimum size of $0.20m$ and maximum face size $0.25m$; for prismatic expansion, a growth rate of 1.2, a transition ratio of 0.77, and a maximum of 3 layers. The wind direction is shown in Figure 6, where the flow streamlines are visualised for both OM (left) and OM+PM (right). Note that, unlike for an isolated building where the wind direction can be easily changed by rotating the model, now with the contextual OM the two are independent of one another.

In the test case (the OM simulation), the wind speed is applied at an upstream inlet with a reference speed (v_r) of $10m \cdot s^{-1}$ at a reference height (z_r) of $10m$. The most commonly used



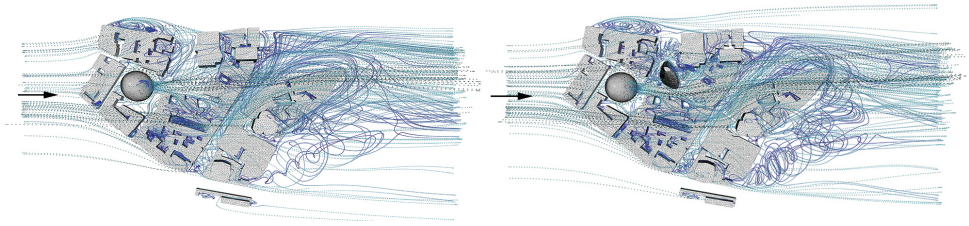
(a) (b)

Figure 5: Simulation domain sizes: (left) OM; (right) PM.

distribution of wind speed with height is the 'power-law' expression:

$$v_x = v_r \cdot (z_x/z_r)^{\alpha} \tag{1}$$

The exponent α is an empirically derived coefficient that is dependent on the stability of the atmosphere. For neutral stability conditions it is approximately 0.143, and is appropriate for open-surroundings such as open water or landscape (Hsu et al., 1994). In the training models a constant vertical wind profile is used, albeit with varying speeds, so as to generate a range of upstream wind speeds across the simulated training set for every vertex, i.e. $v_x = v_r$.



(a) (b)

Figure 6: CFD flow field of (a) OM for testing and (b) OM+PM for validation.

A transient large eddy simulation (LES) could alternatively be used instead of RANS to achieve more accurate and time-dependent peak pressures. However, due to time and resource limitations it was not possible to include a comparison in this study.

3.2 Reduced-Order Model Generation

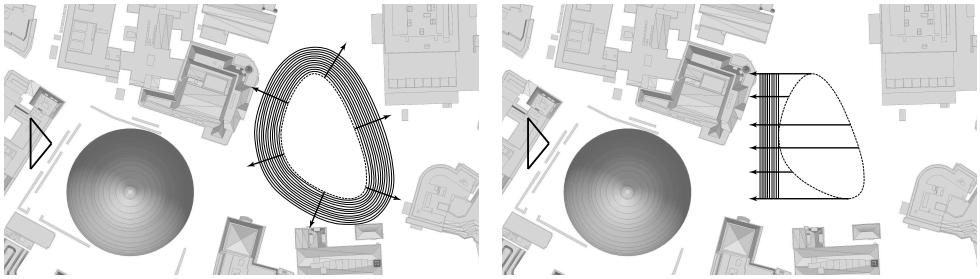
For a training set, $\mathbf{S}$, consisting of vertex feature vectors and simulated pressure extracted from the CFD, the ANN approximates the function $f^{ANN} : \mathbf{X} \rightarrow P$ where $\mathbf{X}$ is the vertex feature vector and P is the vertex pressure. $\mathbf{X}$ is defined as:

$$\mathbf{X}\{v, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{u}\} \tag{2}$$

where $\mathbf{n}_{x,y,z}$ are the vertex normal components; $\mathbf{n}\sigma^{1-5}_{x,y,z}$ are the vertex-ring (one through five) neighbourhood curvature (non-absolute) standard deviation components; and $\mathbf{u}_{x,y,z}$ are the normalised relative vertex position within the model limits. For training, v_S is simply the inlet

wind speed of the training simulation which is constant with height, i.e. no profile. For testing however, this is replaced with v_T the wind speed at the vertex's transformed position in the OM fluid field.

The transformation of the vertex is either a normal offset or a projection upstream from the original location (Figure 7). This results in either $v_{T.offset}$ or $v_{T.upstream}$, both of which are tested for different distances d in the following section. For both transformations, d is increased from 0 at increments of $1m$ until an obstruction is encountered; $12m$ for the normal offset, and $9m$ for the upstream projection.



(a) Offset (b) Upstream

Figure 7: Test feature wind speed location from OM.

From the training feature set, the reduced-order model is generated by a back-propagation artificial neural network (ANN), with a hyperbolic tangent sigmoid transfer function (Vogl et al., 1988):

$$tansig(x) = 2/(1 + exp(-2 \cdot x)) - 1 \tag{3}$$

The ANN structure $X{:}H{:}Y$ is 22:20:1, i.e. 22 input neurons, 20 hidden layer neurons, and 1 output. The sensitivity analysis on the number of neurons in the hidden layer, and the number of layers, is not included here; although 20 in a single layer has been seen to be sufficient. There is generally no rule-of-thumb or guidance to define either, necessitating sensitivity analysis for each problem.

4 Results

The distinction is drawn between the simulation output response Y from CFD, and the prediction output response Y' from the reduced-order model. For a single vertex sample i, the difference between the Y and Y' is used to calculate the sample prediction error, δ_i:

$$\delta_i = (Y_i' - Y_i)/(Y_{max.} - Y_{min.}) \tag{4}$$

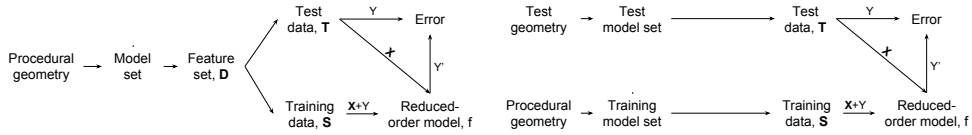
The descriptive statistics used for reporting the errors throughout are:

$\delta_{min.}$	*real-valued minimum of the error range [%]*
$\delta_{max.}$	*real-valued maximum of the error range [%]*
$\overline{\lvert\delta\rvert}$	*mean of the absolute error range [%]* ($\overline{\lvert\delta\rvert} \neq \lvert\bar{\delta}\rvert$)
$\sigma_{\lvert\delta\rvert}$	*standard deviation of the absolute error range [%]* ($\sigma_{\lvert\delta\rvert} \neq \sigma_\delta$)

There are two types of test used here: sample-based (Figure 8a) or model-based (Figure 8b). In the sample-based assessment, test data set $\mathbf{T}$ of size m and training data set $\mathbf{S}$ of size n are drawn from the same set of available data D, meaning that $m = D - n$. Both $\mathbf{T}$ and $\mathbf{S}$ are randomised in this case, and are used to monitor error convergence during the ANN training. For model-based tests, a completely different test set is used so that $\mathbf{T}$ and $\mathbf{S}$ are independently generated, such as in the case where a procedural model is used for training and real models for testing.

The distinction is less clear here since the same geometry is used for both training and testing, i.e. the training geometry is the PM. However, the study is focused on the difference between

the wind speed component in the training and test features. The sample-based test here, more precisely, is purely composed of data from the training simulation; whereas the model-based test derives its data from the test OM simulation.



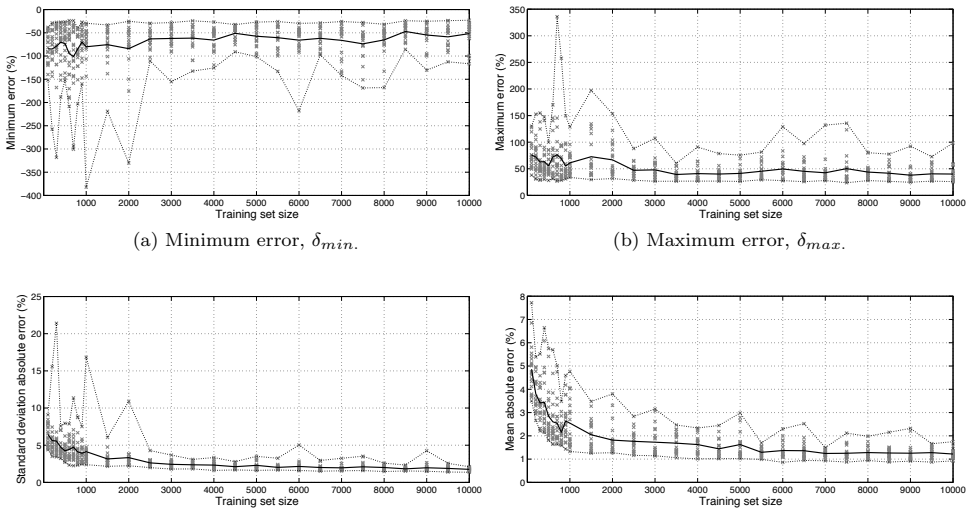
(a) Sample-based (b) Model-based

Figure 8: Testing of sample- and model-based data sets.

Whilst the descriptive statistics for the sample- and model-based accuracy are a good indicator of the ROM's performance, a qualitative visual comparison of simulated and predicted surface pressure is also included at the end.

4.1 Training Set Size (n)

Sample-based errors given are at the converged training set size (Figure 9) of n=10000. The test set size m, being the full data set D minus the training set n, is therefore $210000 - 10000 = 200000$. These are randomly selected for each training run, which is repeated 20 times. The individual runs are shown as grey crosses, with the black lines showing the limits and the blue line the mean over the 20 re-runs. The training set size is increased incrementally, by an increment of 100 from 100 to 1000, and an increment of 500 from 1000 to 10000. The converged sample-based errors are: $\delta_{min.}$ = -51.906%, $\delta_{max.}$ = 40.115%, $\overline{|\delta|}$ = 1.217%, and $\sigma_{|\delta|}$ = 1.756%.



(a) Minimum error, $\delta_{min.}$ (b) Maximum error, $\delta_{max.}$

(c) Standard deviation absolute error, $\sigma_{|\delta|}$ (d) Mean absolute error, $\overline{|\delta|}$

Figure 9: Sample-based training error convergence.

4.2 Training Set Wind Speed (v_S)

Varying the increments of inlet wind speed in the training set simulations has an impact on the time required for initially generating the ROM. The range of wind speeds is kept constant, between 1 and $15m{\cdot}s^{-1}$, and the increments varied between 1, 2, 7, and $14m{\cdot}s^{-1}$. The difference in error between an increment of 1 and $2m \cdot s^{-1}$ is minimal, yet the time saving is substantial with nearly half the number of simulations required. In fact, even with an increment of $7m \cdot s^{-1}$

the difference in error is still minimal, but with a fifth of the time required for generating training data.

Table 3: Error results summary for v_S sensitivity analysis: model-based.

Inc.	v_S $[m \cdot s^{-1}]$	**No. Models**	D	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$
1	{1,2,3,...,15}	15	210000	-57.435	30.801	5.082	7.793
2	{1,3,5,...,15}	8	112000	-66.017	29.099	5.055	7.699
7	{1,8,15}	3	42000	-55.179	28.301	5.673	8.122
14	{1,15}	2	28000	-65.854	25.884	9.995	10.162

Figure 10 shows the probability density distribution of the wind speeds in the test data set from the OM with an offset of $0m$, i.e. no transformation. The probability density distribution uses a smoothing kernel with a normal distribution and a width of $0.02m \cdot s^{-1}$.

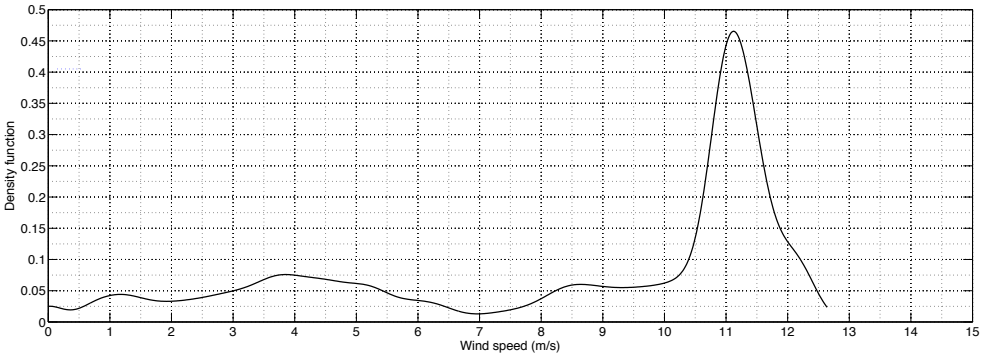


Figure 10: Test set wind speed probability distribution at PM.

A peak at $11.2m \cdot s^{-1}$ is clear due to the inlet wind profile reference speed of $10m \cdot s^{-1}$ at a reference height of $10m$. Further work should establish a methodology for robustly sampling wind speeds: that is, a training set that fits the above distribution would be optimal for this case, but not for another case.

4.3 Test Set Wind Speed Location ($v_{T.offset}$ vs. $v_{T.upstream}$)

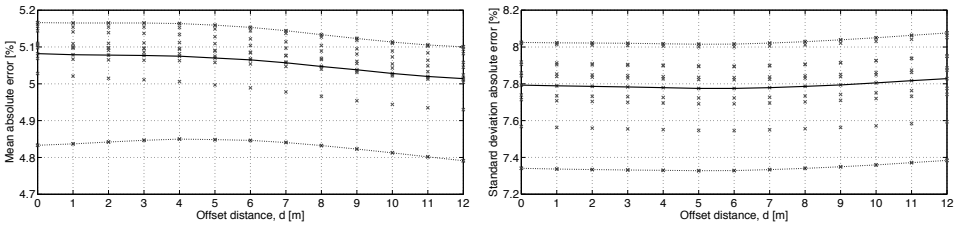
Two transformation methods, normal offset and upstream projection, are tested for varying distances d (Figure 7). The geometric transformation is applied to the PM mesh, positioned in the OM field; from which the wind speeds for the test feature vector, $v_{T.offset}$ or $v_{T.upstream}$, are calculated.

Table 4: Error results summary for test location sensitivity analysis: model-based.

$v_{T.offset}$					$v_{T.upstream}$				
d $[m]$	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	d $[m]$	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$
0	-57.435	30.801	5.082	7.793	0	-57.711	30.793	5.240	7.934
1	-57.404	30.807	5.079	7.789	1	-57.796	30.793	5.232	7.942
2	-57.386	30.811	5.078	7.786	2	-57.744	30.794	5.221	7.946
3	-57.363	30.814	5.077	7.783	3	-57.688	30.794	5.209	7.952
4	-57.342	30.817	5.075	7.779	4	-57.653	30.795	5.203	7.965
5	-57.323	30.819	5.070	7.775	5	-57.669	30.795	5.201	7.981
6	-57.304	30.819	5.065	7.775	6	-57.686	30.795	5.201	7.997
7	-57.286	30.818	5.057	7.779	7	-57.705	30.796	5.204	8.012
8	-57.303	30.813	5.047	7.786	8	-57.747	30.796	5.209	8.027
9	-57.335	30.806	5.038	7.794	9	-57.828	30.796	5.216	8.045
10	-57.386	30.794	5.028	7.805					
11	-57.427	30.773	5.020	7.817					
12	-57.425	30.772	5.014	7.828					

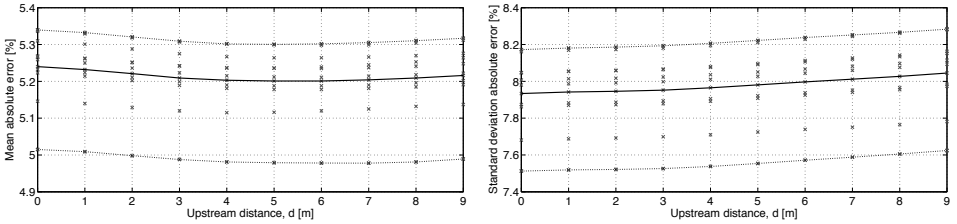
In fact, the prediction errors in Table 4 for both transformation methods with a varying d, suggest they are relatively invariant to the test feature wind speed location. For the offset

there is a standard deviation over the range of absolute mean errors of only 0.024%, and only 0.014% for the upstream. Figures 11 and 12 plot the mean absolute (left) and standard deviation absolute (right) errors against distance for both the offset and upstream transformations. The individual runs are shown as grey crosses, with the black lines showing the limits and the blue line the mean over the 10 re-runs.



(a) Mean absolute error, $\overline{|\delta|}$ (b) Standard deviation error, $\sigma_{|\delta|}$

Figure 11: Test wind speed location: $v_{T.offset}$ distance vs. error.



(a) Mean absolute error, $\overline{|\delta|}$ (b) Standard deviation error, $\sigma_{|\delta|}$

Figure 12: Test wind speed location: $v_{T.upstream}$ distance vs. error.

With the normal offset transformation, the mean absolute error decreases as the offset distance increases so the minimum is at the greatest distance, d=12m. For the upstream projection transformation, the mean absolute error decreases with distance until d=5 or 6m, at which point it starts to increase again.

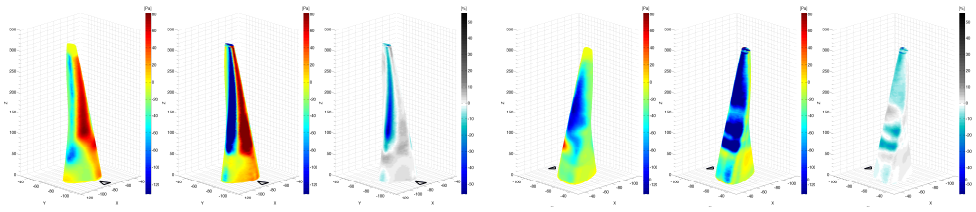
As mentioned, the variation of error with distance or transformation method is small so the optimal location at which to measure the wind speed for the test feature is still unclear. However, it is likely to be esoteric for each problem or OM and should be studied further.

4.4 Model-Based Pressure Distribution

The following model-based results use n=10000, a training wind speed increment of $1m \cdot s^{-1}$, and no transformation on the test feature wind speed location. The model-based errors are: $\delta_{min.}$ = -57.435%, $\delta_{max.}$ = 30.801%, $\overline{|\delta|}$ = 5.082%, and $\sigma_{|\delta|}$ = 7.793%. Figures 13 and 14 visualise the surface pressures and errors on the PM.

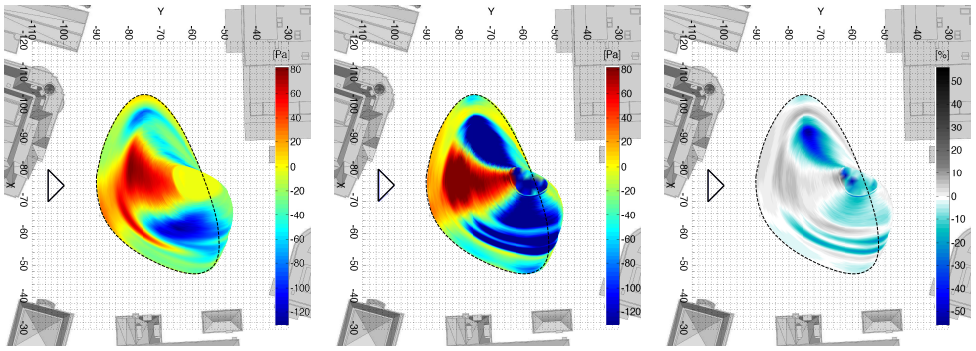
In Figure 13, the left three figures are from an upstream perspective; the right three from downstream. Within each triplet: the left figure is the CFD simulation (Y $[Pa]$), centre the ANN prediction (Y' $[Pa]$), and right the difference (δ [%]) between predicted and simulated pressures. The inlet wind direction is indicated by an arrow on each figure.

There is generally good agreement in the spatial distribution of positive and negative pressure between the simulated and predicted models. Although the localised over-prediction (i.e. $|Y'| > Y$) of positive and negative pressure values, relative to the simulated values, is apparent in an exaggeration of the visual results. The probability distribution, or density function, of non-absolute errors [%] for 10 re-runs (grey) and their mean (black) are shown in Figure 15. A training set of 10000, an increment of $1m \cdot s^{-1}$ in the training set wind speeds, and no test feature wind speed location transformation are applied.



(a) CFD (b) ROM (c) Error (d) CFD (e) ROM (f) Error

Figure 13: Model-based test: (a-c) upstream; (d-f) downstream side.



(a) CFD (b) ROM (c) Error

Figure 14: Model-based test: plan view.

A smoothing kernel with a normal distribution and a width of 0.01% is used. The errors fit a normal distribution, with a peak probability of about 19% that the error will be 0.6%. Towards the minimum and maximum of the error range (-10 and 10%), the probability is only between 1.5 and 0% respectively.

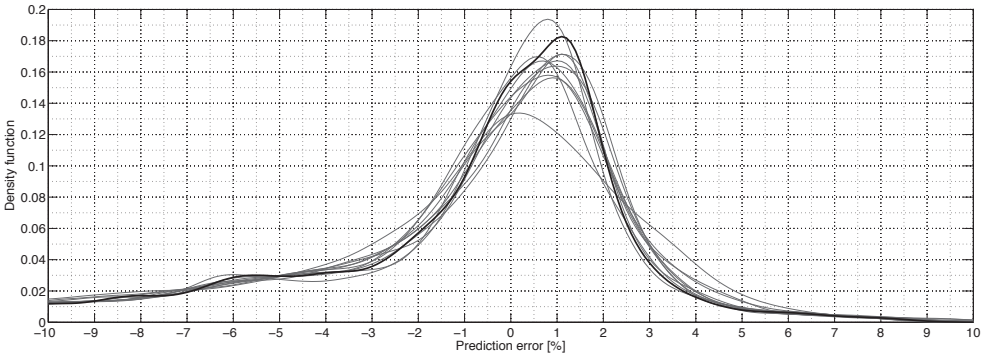


Figure 15: Probability distribution of prediction errors.

5 Discussion

These developments represent an alternative approach that is fundamentally different to previous attempts at interference generalisation found in the literature. The use of local features rather than global parameters allows for arbitrary complexity in the obstruction model and for vertex surface pressure visualisation rather than the global interference factor.

Compared to solver approximation techniques, such as the FFD solver, solution approximation has the benefit of being based on a conventional, higher accuracy CFD solver. As such, the validity of the basis data can, to a larger extent, be trusted or verified. The comparative disadvantage is that the FFD can produce field rather than surface data which is useful for identifying flow patterns, assessing pedestrian comfort, and to gauge the secondary downstream effects that a new building will have on others.

Three sensitivity analyses were run on the training set size, the training set wind increment, and the test set wind speed location. The first found that a sample size of 10000 was adequate to reach error convergence during the ANN training. Secondly, the number of training simulations can be reduced safely from an increment of $1m\cdot^{-1}$ to 2 without much effect on the result, or even to 7 with still a reasonable effect considering the time saving. And thirdly, the results were found to be mostly invariant to the test set wind speed location, i.e. the transformation of the PM in the OM field.

Following these, the final model-based test was visualised to check the predicted pressure distribution qualitatively against the simulation. Generally, over-prediction can be seen, but general patterning or distribution of both positive and negative pressure remains intact.

5.1 Process Time Analysis

The feature extraction times are based on a calculation speed of $0.02784s/sample$, for n=10000 off-line ROM generation and m=14000 on-line prediction. And the number of PM models in the training set is 15, although it has been shown that this can be reduced to 8 or even 3 without significant reduction in accuracy. The PM simulation time is $1517s$ (28minutes) per model. The ANN training time, for n=10000, is averaged over 20 runs; the mean time is $38.269s$ (σ:$17.143s$).

Table 5: CFD and ROM process times.

Conventional CFD Process		**Test time** $[s]$
PM + OM simulation	**Total**	10709
Off-line ROM Process		**Test time** $[s]$
PM simulations (15no.)	22755	
OM simulation	10080	
Feature extraction	278.4	
ANN training	38.269	
	Total	33151.669
On-line ROM Process		**Test time** $[s]$
PM feature extraction	389.76	
PM prediction	0.023	
	Total	389.783

The model-based prediction times show that, in comparing only on-line processes, the ROM is 27.47-times faster than the conventional CFD method. However, this does not take into account the full process. By comparing the off-line plus on-line processes for repetition, where x is the number of design iterations, the CFD time=$10709x$ and the ROM time=$389.783x$+33151.669 (Figure 16). In solving for x, the minimum number of iterations before the full ROM process time equals the CFD is x=3.21.

5.2 Limitations

It is key to stress the primary limitation of this work is the similarity between the training and test geometry, i.e. the training model is the same as the PM. The paper is therefore constrained in the conclusions it can draw without subsequent testing. Further improvements and generalisation could be made by generating training shape features from a procedural model. This remains to be tested due to the combinatorial problem of sampling a procedural model under a large range of wind speeds. For instance, Wilkinson et al. (2013) shows that a set of 400 procedural models evaluated with RANS could be used to make predictions on isolated tall building models. Following that method, the training set would be up to 400 multiplied by the number of training wind speed iterations, i.e. 3, 8, or 15 simulations per procedural model instance. Although this would require substantial investment, it is an off-line process which would only be required once.

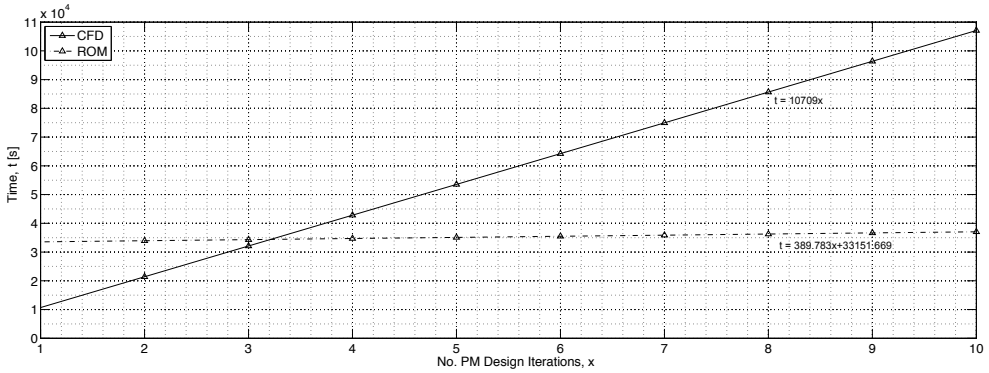


Figure 16: Process time t against number of design iterations x for CFD and ROM.

Further testing on alternative geometries and optimisation of the feature vector and its calculation will likely improve robustness, accuracy, and speed. The benefits of the ROM also increase with the cost of the basis simulation, for instance if a more costly RANS or LES is used.

6 Conclusion

In summary, the methodology and results presented here demonstrate an alternative approach to urban wind interference approximation for tall building design. The results indicate that significant improvements in response time (27-times faster when comparing on-line prediction times with conventional CFD) can be made with a reasonable trade-off in accuracy (mean absolute errors of around 5.0% σ:7.8%). Sensitivity analyses suggested that: i) a training set size of n=10000 samples was adequate; ii) wind speed increments of the training simulations v_S can be increased to 2 or even $7m \cdot s^{-1}$ without great effect, although it may potentially affect the generalisability to other scenarios; and iii) the accuracy with the test set wind speed location v_T is relative invariant to both offset or upstream transformation and distance d. Although there remain limitations to our approach before use in practice, predominantly the applicability to varying PM geometry, we believe that in conjunction with previous work it represents a significant step towards interactive building design via reduced-order wind interference modelling.

References

Agrawal, N., Mittal, A. K., & Gupta, V. K. (2012). Along-Wind Interference Effects on Tall Buildings. In National Conference on Wind Engineering, India (pp. 193–204).

ANSYS. (2009). CFX-Solver Theory Guide (Tech. Rep.). Canonsburg, PA: ANSYS Inc.

ANSYS. (2014). CFX v13.0. Retrieved 05.02.2014, from www.ansys.com

Athanailidi, P., Fatah gen Schieck, A., Tenu, V., & Chronis, A. (2014). Tensegrity Systems Acting as Windbreak: Form Finding and Fast Fluid Dynamics Analysis to Address Wind Funnel Effect. In Proceedings of SimAUD SCS SpringSim'14 (pp. 127–134). Tampa, Fl..

Bailey, P. A., & Kwok, K. C. S. (1985). Interference Excitation of Twin Tall Buildings. *Journal of Wind Engineering and Industrial Aerodynamics*, *21*, 323–338.

Bentley Systems. (2013). GenerativeComponents v08.11.08.296. Retrieved 01.02.2014, from www.bentley.com

Bui-Thanh, T., Willcox, K., & Ghattas, O. (2008). Parametric Reduced-Order Models for Probabilistic Analysis of Unsteady Aerodynamic Applications. *AIAA*, *46*(10).

Chittka, L., Skorupski, P., & Raine, N. E. (2009). Speed-Accuracy Tradeoffs in Animal Decision Making. *Trends in Ecology & Evolution*, *24*(7), 400–407.

Chronis, A., Tsigkari, M., Davis, A., & Aish, F. (2012). Design Systems, Ecology and Time. In Proceedings of ACADIA12: Synthetic Digital Ecologies. San Francisco, CA.

Chronis, A., Turner, A., & Tsigkari, M. (2011). Generative Fluid Dynamics: Integration of Fast Fluid Dynamics and Genetic Algorithms for Wind Loading Optimization of a Free Form Surface. In Proceedings of SimAUD SCS SpringSim'11 (pp. 79–86).

Degroote, J., Vierendeels, J., & Willcox, K. (2010). Interpolation Among Reduced-Order Matrices to Obtain Parameterized Models for Design, Optimization and Probabilistic Analysis. *International Journal for Numerical Methods in Fluids*, *2010*(63), 207–230.

English, E. C., & Fricke, F. R. (1999). The Interference Index and its Prediction using a Neural Network Analysis of Wind-Tunnel Data. *Journal of Wind Engineering and Industrial Aerodynamics*, *83*, 567–575.

Gosnell, M. R., Woodley, R. S., & Gorrell, S. E. (2012). Results of Mining Data Features During Computational Fluid Dynamics Simulations (Tech. Rep.). Provo, UT: Brigham Young University.

Graening, L., Menzel, S., Hasenjäger, M., Bihrer, T., Olhofer, M., & Sendhoff, B. (2008). Knowledge Extraction from Aerodynamic Design Data and its Application to 3D Turbine Blade Geometries. *Journal of Mathematical Modelling and Algorithms*, *7*(4), 329–350.

Graening, L., & Sendhoff, B. (2014). Shape Mining: A Holistic Data Mining Approach for Engineering Design. *Advanced Engineering Informatics*.

Gu, M., & Xie, Z.-N. (2011). Interference Effects of Two and Three Super-Tall Buildings under Wind Action. *Acta Mechanica Sinica*, *27*(5), 687–696.

Hsu, S. A., Meindl, E. A., & Gilhousen, D. B. (1994). Determining the Power-law Wind-Profile Exponent under Near-Neutral Stability Conditions at Sea. *Journal of Applied Meteorology*, *33*(1994), 757–772.

Jianguang, Z. (2008). Interference Effects on Wind Loading of a Group of Tall Buildings in Close Proximity. Unpublished doctoral dissertation, The University of Hong Kong.

Karagkouni, C. S., Fatah, A., Tsigkari, M., & Chronis, A. (2013). Façade Apertures Optimization : Integrating Cross-Ventilation Performance Analysis in Fluid Dynamics Simulation. In Proceedings of SimAUD SCS SpringSim'13.

Khanduri, A. C. (1997). Wind-Induced Interference Effects on Buildings - Integrating Experimental and Computerized Approaches. Unpublished doctoral dissertation, Concordia University, Montreal, CA.

Khanduri, A. C., Bedard, C., & Stathopoulos, T. (1997). Modelling Wind-Induced Interference Effects using Backpropagation Neural Networks. *Journal of Wind Engineering and Industrial Aerodynamics*, *72*, 71–79.

Khanduri, A. C., Stathopoulos, T., & Bedard, C. (1998). Wind-Induced Interference Effects on Buildings - A Review of the State-of-the-Art. *Engineering Structures*, *20*(7), 617–630.

Lam, K. M., Zhao, J. G., & Leung, M. Y. H. (2011). Wind-Induced Loading and Dynamic Responses of a Row of Tall Buildings under Strong Interference. *Journal of Wind Engineering and Industrial Aerodynamics*, *99*(5), 573–583.

Lee, B. E., & Fowler, G. R. (1975). The Mean Wind Forces Acting on a Pair of Square Prisms. *Building Science*, *10*, 107–110.

Lu, S. C.-Y., Tcheng, D. K., & Yerramareddy, S. (1991). Integration of Simulation, Learning and Optimization to Support Engineering Design. *Annals of the CIRP*, *40*(1), 143–146.

Malkawi, A. M. (2004). Developments in Environmental Performance Simulation. *Automation in Construction*, *13*(4), 437–445.

Post, F. H., Vrolijk, B., Hauser, H., Laramee, R. S., & Doleisch, H. (2002). Feature Extraction and Visualisation of Flow Fields. In Eurographics.

Ramanathan, S., & Graening, L. (2009). Knowledge Incorporation into Evolutionary Algorithms to speed up Aerodynamic Design Optimizations Soundappan Ramanathan. Unpublished doctoral dissertation, Universitat Stuttgart.

Rendall, T. C. S., & Allen, C. B. (2008). Multi-Dimensional Aircraft Surface Pressure Interpolation using Radial Basis Functions. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, *222*(4), 483–495.

Sakamoto, H., & Haniu, H. (1988). Aerodynamic Forces Acting on Two Square Prisms placed Vertically in a Turbulent Boundary Layer. *Journal of Wind Engineering and Industrial Aerodynamics*, *31*, 41–66.

Saunders, J. W., & Melbourne, W. H. (1979). Buffeting Effects of Upwind Buildings. In Proceedings of the 5th International Conference on Wind Engineering (pp. 593–605). Fort Collins, CO.

Schilders, W. (2008). Introduction to Model Order Reduction. In Model Order Reduction: Theory, Research Aspects and Applications (13th ed.). Springer.

Srinivasan, R. S., & Malkawi, A. M. (2004). The Use of Learning Algorithms for Real-Time Immersive Data Visualisation in Buildings. In SIGRADI (pp. 329–332).

Stam, J. (1999). Stable Fluids (Tech. Rep.). Seattle: Alias-Wavefront.

Stathopoulos, T. (1984). Adverse Wind Loads on Low Buildings Due to Buffeting. *Journal of Structural Engineering*, *110*(10), 2374–2392.

Taniike, Y. (1992). Interference Mechanism for Enhanced Wind Forces on Neighboring Tall Buildings. *Journal of Wind*, *42*, 1073–1083.

Taniike, Y., & Inaoka, H. (1988). Aeroelastic Behavior of Tall Buildings in Wakes. *Journal of Wind Engineering and Industrial Aerodynamics*, *28*(1-3), 317–327.

Vogl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., & Alkon, D. L. (1988). Accelerating the Convergence of the Back-propagation Method. *Biological Cybernetics*, *59*(4-5), 257–263.

Wilkinson, S., Bradbury, G., & Hanna, S. (2014). Approximating Urban Wind Interference. In Proceedings of SimAUD SCS SpringSim'14. Tampa, FL..

Wilkinson, S., Hanna, S., Hesselgren, L., & Mueller, V. (2013). Inductive Aerodynamics. In Proceedings of eCAADe 2013: Computation and Performance. Delft, NL..

Xie, Z.-N., & Gu, M. (2004). Mean Interference Effects among Tall Buildings. *Engineering Structures*, *26*(9), 1173–1183.

Zhang, A., Gao, C., & Zhang, L. (2005). Numerical Simulation of the Wind Field around Different Building Arrangements. *Journal of Wind Engineering and Industrial Aerodynamics*, *93*(12), 891–904.

Zhang, A., & Gu, M. (2008). Wind-Tunnel Tests and Numerical Simulations of Wind Pressures on Buildings in Staggered Arrangement. *Journal of Wind Engineering and Industrial Aerodynamics*, *96*(10-11), 2067–2079.

Zhang, A., & Zhang, L. (2004). RBF Neural Networks for the Prediction of Building Interference Effects. *Computers & Structures*, *82*(27), 2333–2339.

Zuo, W., & Chen, Q. (2009). Real-Time or Faster-than-Real-Time Simulation of Airflow in Buildings. *Indoor Air*, *19*(1), 33–44.

Zuo, W., & Chen, Q. (2010). Fast and Informative Flow Simulations in a Building by using Fast Fluid Dynamics Model on Graphics Processing Unit. *Building and Environment*, *45*(3), 747–757.

A.2 Wilkinson, S. and Hanna, S. (2014). Approximating Computational Fluid Dynamics for Generative Tall Building Design. *International Journal of Architectural Computing*, 12(2), pp.155-178.

Approximating Computational Fluid Dynamics for Generative Tall Building Design

Samuel Wilkinson and Sean Hanna

international journal of architectural computing **issue 2, volume 12** | **155**

Approximating Computational Fluid Dynamics for Generative Tall Building Design

Samuel Wilkinson and Sean Hanna

Background literature review, methodology, results, and analysis are presented for a novel approach to approximating wind pressure on tall buildings for the application of generative design exploration and optimisation. The predictions are approximations of time-averaged computational fluid dynamics (CFD) data with the aim of maintaining simulation accuracy but with improved speed. This is achieved through the use of a back-propagation artificial neural network (ANN) with vertex-based shape features as input and pressure as output. The training set consists of 600 procedurally generated tall building models, and the test set of 10 real building models; for all models in both sets, a feature vector is calculated for every vertex. Over the test set, mean absolute errors against the basis CFD are 1.99–4.44% (σ:2.10–5.09%) with an on-line process time of 14.72–809.98*s* (0.028*s*/*sample*). Studies are also included on feature sensitivity, training set size, and comparison of CFD against prediction times. Results indicate that prediction time is only dependent on the number of test model vertices, and is therefore invariant to basis CFD time.

1. INTRODUCTION

Although computational fluid dynamics (CFD) has existed now for over 50 years and parametric CAD for over 30, both have seen an increased interest in architectural practice over the last decade. However, in computational design, especially in generative exploration or optimisation, CFD remains a challenging simulation tool to integrate. There are at least three reasons for this: firstly, the cost of expertise and software is high; secondly, the relationship between a design and its fluid environment is complex, often subtle, and esoteric; and thirdly, the time required to achieve accurate results is typically greater than that available, namely at early project stages when the guidance provided by the simulation is most valuable.

The third issue is fundamentally one of approximation, a trade-off common in simulation of time against accuracy. This relationship is marked by the two characteristic extremes of fast-inaccurate and slow-accurate, with a range of solutions existing along this spectrum. Responses to this problem can be categorised into two forms: i) type-one is solver approximation, including all conventional CFD methods which by one approach or another seek to emulate the full underlying physical fluid behaviour. Any approach of this type can only fit within the time-accuracy spectrum since the two properties remain dependent; ii) type-two is solution approximation, encompassing methods which aim to emulate simulation behaviour. Through model reduction and machine learning, a break from the established trade-off can enable a move towards fast-yet-accurate approaches.

Effects of the wind upon buildings are numerous: for pedestrian comfort in surrounding proximity; ventilation and therefore thermal comfort and indoor air quality; and structural performance. Wind loads, along with seismic, are the two primary external forces that increase with building height. Therefore tall buildings have been identified as a focal typology for this and a number of reasons. The aerodynamic shape has a primary impact on these forces and therefore subsequently on the overall structural, material, energy, and financial performance.

The trend has always been to build them as high as (contextually, economically and structurally) possible, necessitating cutting-edge design and construction technologies. With the quantity, height, and complexity of tall buildings still increasing, there is a greater need for early-stage form analysis and optimisation. The geometric complexity in the latest generation has broken away from the previously necessary extruded planform to more freeform shapes. This has been facilitated by the recent ubiquity of computation in design, analysis, fabrication, and construction. Tall buildings generally lend themselves well to parametric design since there is often a strong vertical repetition which can be expressed easily computationally. It also means it is suitable for generating procedural models that, with a

relatively small number of parameters, can represent a large number of potential designs.

1.1. Contribution

The aim of this paper is to demonstrate a new approach to approximating wind pressure on tall buildings for the application of generative design exploration and optimisation. The predictions are approximations of time-averaged computational fluid dynamics (CFD) data with the aim of maintaining simulation accuracy but with improved speed. This is achieved through the use of a back-propagation artificial neural network (ANN) with vertex-based shape features as input and pressure as output. Success of the approach is measured against the objective of being fast-yet-accurate; therefore the time and errors are quantified in the end discussion.

2. LITERATURE REVIEW

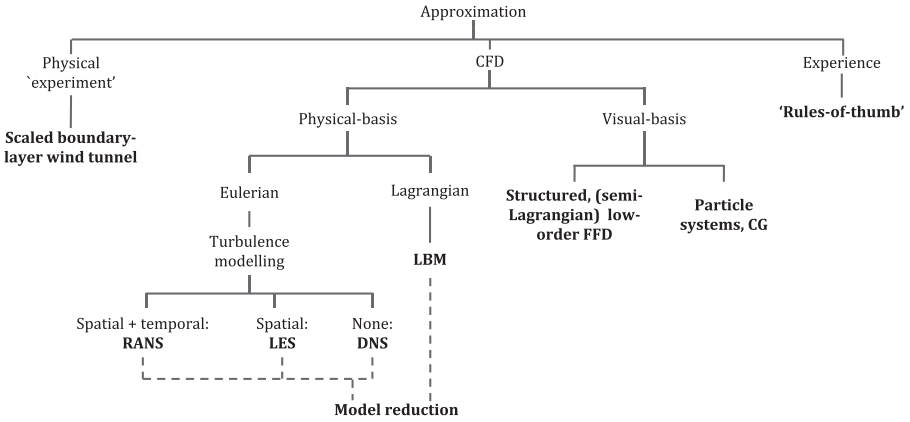
The background review of existing literature is divided into three parts: i) solver approximation; ii) solution approximation; and iii) shape features.

2.1. Solver approximation

Most approaches towards CFD approximation focus on simplification of the solver itself. For instance: simplified meshes (spatial discretisation); the use of lower-order equations; or the treatment of turbulence through modelling. These methods can be classed as type-one, *solver approximation* (Figure 2.1). For instance, RANS (Reynolds-Averaged Navier-Stokes), LES (Large Eddy Simulation), and DNS (Direct Numerical Simulation) all treat turbulence with different numerical approaches, i.e. temporally, spatially, and directly.

Another example is the 'Stable Fluids' fast fluid dynamics (FFD) solver developed by [1] for the computer graphics and games industries, which has subsequently been developed and tested for architectural applications [2, 3, 4, 5]. Development and application for architectural design was motivated by three factors: a validation study suggested it as suitable for purpose, even though it was limited to indoor, low Reynolds number flow regimes [6, 7]; the qualitative appearance of accuracy for turbulent flows; and its remarkable speed compared to traditional CFD methods like RANS (Reynolds-Averaged Navier-Stokes). [6] implemented the FFD with a zero-equation turbulence model but found that it performed worse since it was not designed or suited to the FFD approach. It should be noted, however, that with a lack of turbulence model, the solver relies on continuous interaction (such as game character movement) to compensate for numerical dissipation.

Although such recent developments aim to increase the speed of CFD, they do so at the direct expense of accuracy; the opposite of traditional CFD development where accuracy is increased at the expense of speed.



► Figure 2.1: CFD solver approximation taxonomy.

The benefit of solver over solution approximation is the availability of full spatial field data for all fluid properties, although in some cases such as the FFD, production of surface data is more difficult due to the structured mesh approximation (voxelisation).

2.2. Solution approximation

Another possible approach to this problem, type-two, is *solution approximation*. CFD originated in aeronautics and astronautics, as such there is a large quantity of work directed towards modelling and optimisation of aerofoils, fuselages, and turbine blades. An optimisation routine will often generate large data sets of simulation data, from which knowledge of the problem can be extracted. The following fall into the greater category of supervised machine learning approaches where the relationship between an input feature vector extracted from some geometry and the ground truth data output from full CFD simulation is learnt.

In one such approach, a large model set of turbine blades is used with a decision tree to analyse the relationship between point deformation of models and their change in surface pressure [8, 9]. Areas of high sensitivity can then be mapped onto a pre-defined base geometry and used to focus subsequent analysis. Extension of this work incorporates an evolutionary optimisation process, so as to use the information extracted from previous cases to create non-random initial populations of solutions and to guide the evolution.

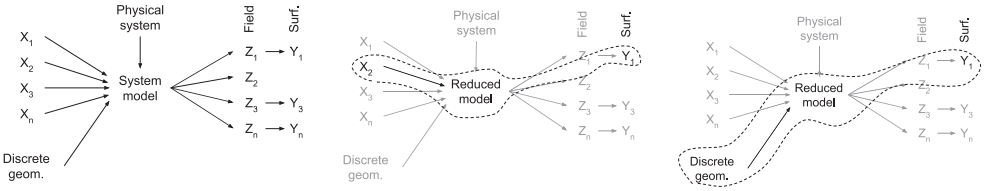
Analyses that are potentially obstructive to the design process may involve partial differential equations (PDEs), such as the Navier-Stokes equations of fluid flow and the Maxwell equations in electromagnetism [10]. Whilst simulation of these phenomena can give high-accuracy results, they are computationally expensive and cannot be computed in real-time. As a

result, a design process using high-accuracy techniques has inherently slow response times and loses any desired interactivity.

Significant efforts have been made to reduce the complexity of these systems in order to make them interactive or suitable for optimisation; this is generally referred to as *model reduction*. Reduced-order models (ROMs) approximate representations of system behaviours, namely for computational simulations with slow response times, with the aim to create a lower-dimensional system model whilst retaining predictive fidelity [11, 12]. They typically do so by restricting the input quantities to boundary conditions and outputs to those of interest (e.g. lift, drag, or a quantity at a single point).

In one example, [13] use spatial and behavioural parameters as input features to a radial basis function (RBF). The RBF is used to interpolate and merge CFD and wind-tunnel data on pressure coefficient values (lift and drag) for aerofoil analysis. They use an input feature vector *Cp{x,y,z, a, M ,Re):* where *x, y, z* is the spatial position; *a* the angle of attack; *M* the Mach number; and Re the Reynolds number.

▼ Figure 2.2: Reduced-order model schematic.



(a) Simulation (b) Typical ROM from boundary conditions (c) Proposed ROM from geometry

Whilst this method proved successful for linking behavioural characteristics *(a, M,* and Re) to data sources (CFD and wind-tunnel), it is limited to a single geometry, thus the use of explicit spatial positions *(x, y,* and *z).* For cases of differing geometry between training and testing, spatial positions become non-unique and can therefore not be used within the feature vector. This necessitates the use of either explicit global design parameters or implicit local shape description.

Using spatial positions (or mesh node numbers) for a feature vector is also proposed by [14]. In this case, an ANN is used to predict post-processed CFD data for rapid visualisation and interpolation of boundary conditions with an augmented reality user display system. The input feature vector, **X,** and output response, **Y,** are defined as: ***X****{n,* S, *P}*, ***Y****{T, V},* where T is the air temperature and V is the air speed at a node, *n* is the mesh node number (1224 nodes in the cubic room), S is the supply temperature, and *P* is the supply pressure. Again the proposal is strongly limited to not only a single geometry but to a single mesh by the use of spatial positions or node

numbers corresponding consistently with fixed locations. The limitation again is that for differing training/test geometry the positions are non-unique.

2.3. Shape features

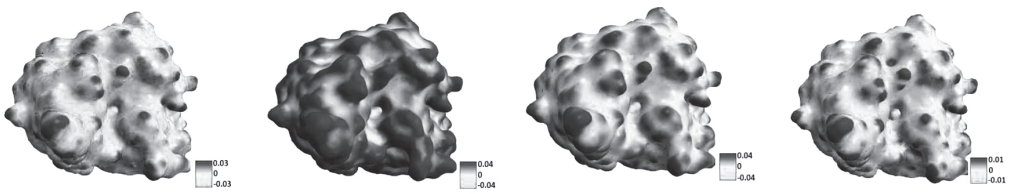
A component of the methodology involves a localised description of a sample point or vertex, constituting the input vector for the machine learning. These are predominantly shape or topological characteristics, although this can be extended to local fluid flow properties. With discretised surface representations (meshes) there is often a need to describe local shape features for a broad range of applications.

An exemplar case is for scale-invariant surface descriptors for the matching of molecular surface regions to identify potential chemical functionality, i.e. binding molecules often have locally complementary shapes [15, 16]. When calculating surface curvature, the distance or neighbourhood size must be included, shown in Figure 2.3. From left to right the features are: minimum curvature; maximum curvature; mean curvature; and Gaussian curvature.

There are a large number of similar studies on mesh curvature, edge detection, and invariant shape analysis, for example [17, 18, 19, 20, 21, 22, 23]. Applications range from chemistry, to rationalising and reconstructing 3-D scanned models, to identifying constant features in images for camera stabilisation.

Whilst [9] do not learn the function between local shape features and pressure to make predictions on new cases, the ability, however, to generate the sensitivity at a point from its deformation is an inspiration for two key elements of this work. Firstly, it highlights the importance of showing the pressure distribution over the entire model rather than calculating a global metric of a design's success, i.e. considering the problem locally rather than globally. Secondly, the use of top-down models (those found in parametric models with global variables) have an inherently limited flexibility. They can be adjusted infinitely within the bounds of the logic of the model parameters, however it is firstly difficult to alter these foundations at later stages and secondly each model will have a different logic, variables, or set of dependencies. It is therefore difficult to use these global variables (for example, a parametric tall building may have *Height* and *Taper Factor* as two of its defining variables) as input features to learning. If they are used, the

▼ Figure 2.3: Shape analysis for protein shape matching in biochemistry [16].



problem being learnt is restricted to that logic. [9], on the other hand, use a local mesh vertex deformation as input feature. Since all CFD simulations require a surface mesh of the geometry to be generated, it is a relatively simple generalisation to use the mesh data and its derivates as input for the learning.

Further generalisation of the method is proposed, specifically for automobile design, in particular for the detection of design novelty or for characterising families of similar products [8]. The key similarity with this work is the use of unstructured surface meshes as the basis geometric representation, which in itself is a good foundation for generating training data due to the high acceptance in design practice. The distinction however is in the definition of the actual learning process and feature vector: [8] use a deformation metric from a base case; as opposed to the here proposed broader shape description.

Their proposed shape mining process focuses on the extraction of performance data from conventional analysis processes for compilation of a large database. From which a meta-representation, or reduced-order model, can be created and used for sensitivity analysis, concept retrieval, and interaction analysis. These data modelling and knowledge formation components link back holistically to knowledge utilisation and decision making processes (DMPs).

3. METHODOLOGY

The approach can be split into the following steps: i) training set procedural geometry generation; ii) training set CFD evaluation; iii) training set feature calculation; iv) ANN training; v) test set geometry generation; vi) test set CFD evaluation; and vii) prediction and assessment.

3.1. Procedural training geometry

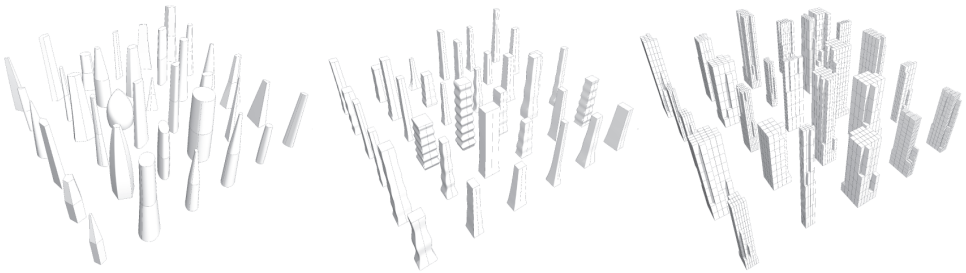
The parametric model was created in *GenerativeComponents* [24]. The goal was to create a generalised tower model, with the two properties of minimising the number of parameters used whilst maximising the design representation potential, i.e. the number of possible buildings it could create. This is important when considering optimisation or exploratory design space searches to avoid the curse of dimensionality. This means that as the number of variables increases, the design space increases exponentially by n^D, where n is the number of samples taken per parameter and D is the number of parameters, or dimensionality. There is therefore clearly a compromise to be made between model efficiency and representability.

The geometry for the training set was generated using a procedural tall building model with a select number of key parameters. There are in fact three separate topologies in the procedural model with their own parameters, since it is difficult to incorporate the entire design space with

one parametric logic [25, 26]. Using the unstructured triangulated surface mesh from these means we are not limited by a single parametric topology in the learning phase of the method [9]. Local surface-mesh shape characteristics are used as input features to the learning algorithm instead of the design parameters, avoiding reliance on any one parametric model definition.

Contemporary tall building design, as discussed previously, has been freed up by computational design tools, analysis, and construction methods. The iconicity of skyscrapers is also a driving force for unique, bespoke forms; as such, increasingly complex forms are being planned and constructed. This presents an interesting challenge to wind engineers who, for the new generation of tall buildings, typically struggle to find general behavioural rules akin to the top-down global learning approach, i.e. bespoke designs require bespoke analysis. In a similar way, a procedural tall building model is used to generate the reduced-order model and is tested on an extended set of 10 real buildings.

The geometry for the training set was generated using a procedural tall building model with a select number of key parameters. There are three separate topologies in the procedural model each with their own parameters, shown in Figure 3.1.



(a) Extrusion (b) Periodic (c) Blocks

▲ Figure 3.1: Procedural training model sets.

The three procedural models can be classified as one of three topologies: *Extrusion*; *Periodic*; and *Blocks*. The topology is initially selected randomly, and then each can be used to generate instances by randomly assigning parameter values from the ranges given in Table 3.1.

With these parameter sets and ranges, the maximum number of potential instances for the three topologies respectively is: $2.56e^{16}$; $2.34e^{21}$; and $3.24e^{12}$; giving a sum of the three of $2.3448e^{21}$. Although this is the maximum number, certain combinations or regions of the parameter space lead to invalid instances which reduces the total. These are filtered out in the code simply with a *while(solid.Success==false)* statement. A training set of 600 instances is generated, giving a sampling of $2.56e^{-17}$ % of the total parameter space.

	N	W	D	H	mSF	tSF	F	o	R	A	D	f	fO	Nw	Nd	Nh
Extrusion																
Min.	3	10	10	100	0.5	0.1	0.3	2	0	-	-	-	-	-	-	-
Max.	7	60	60	200	1.1	1.1	5.1	4	180	-	-	-	-	-	-	-
Inc.	1	0.1	0.1	0.1	0.1	0.1	0.1	1	0.1	-	-	-	-	-	-	-
Periodic																
Min.	-	10	10	100	-	-	0.5	-	0	0.1	100	0.1	-180	-	-	-
Max.	-	20	20	200	-	-	20	-	180	10	1000	2	180	-	-	-
Inc.	-	0.1	0.1	0.1	-	-	0.1	-	0.1	0.1	1	0.1	1	-	-	-
Blocks																
Min.	-	10	10	100	-	-	-	-	0	-	-	-	-	4	4	5
Max.	-	60	60	200	-	-	-	-	180	-	-	-	-	7	7	15
Inc.	-	0.1	0.1	0.1	-	-	-	-	0.1	-	-	-	-	1	1	1

Note: *N* no. edges; *W* width [*m*]; *D* depth [*m*]; *H* height [*m*]; *mS F* mid planform scale factor; *tS F* top planform scale factor; *F* fillet radius [*m*]; *o* planform curvature order; *R* orientation [°]; *A* amplitude; *D* decay; *f* frequency; *f O* frequency offset; *Nw* no. blocks in width; *Nd* no. blocks in depth; *Nh* no. blocks in height.

▲ Table 3.1: Procedural training model parameter ranges.

3.2. CFD simulation

CFX 13.0 [27] is used for the steady-state time-averaged Reynolds-Averaged Navier-Stokes (RANS) simulations with a k-∈ turbulence model. Typically the models are meshed with roughly an equal number of cells (up to the maximum available computational resources), of around four million elements. Each simulation, depending on the complexity, requires on average 1914.470*s* (σ:629.808*s*) to converge on a 2.66*GHz i*7 4GB RAM.

For the ground, a no slip smooth wall is assigned (i.e. fluid velocity at wall boundary is zero); for the sides and top parallel to the flow, a free slip wall (i.e. zero shear stress from wall friction); and for the outlet, a zero relative pressure opening. For the inlet, the wind profile is applied as described below, with a medium intensity turbulence and eddy viscosity ratio [28].

Basic simulation parameters are: high-resolution advection and turbulence numerics; isothermal fluid at 25°*C*; a scalable wall function; and a convergence residual target of $1.0e^{-6}$ RMS. The following meshing parameters are used: an unstructured tetrahedral domain mesh, with patch independence; a boundary surface element size of 5*m*; a model surface minimum size of 0.20*m* and maximum face size 0.25*m*; for prismatic expansion, a growth rate of 1.2, a transition ratio of 0.77, and a maximum of 3 layers.

The wind speed is applied at an upstream inlet with a reference speed (v_r) of $10m \cdot s^{-1}$ at a reference height (z_r) of 10*m*. The most commonly used distribution of wind speed with height is the 'power-law' expression:

$$v_x = v_r \cdot (z_x/z_r)^{\alpha} \quad (1)$$

The exponent α is an empirically derived coefficient that is dependent on the stability of the atmosphere. For neutral stability conditions it is

approximately 0.143, and is appropriate for open-surroundings such as open water or landscape [29].

Both the training (Figure 3.2) and test sets are evaluated under the same boundary conditions and with the same wind profile. The 'static' pressure [*Pa* or $N \cdot m^{-2}$] is the force per unit area, taken at sample-points (vertices) over the model surfaces (mesh).

3.3. Shape feature vector

The basic concept is to define the pressure at a point or vertex on a model by its geometric characteristics, i.e. its relative position on the model, proximity to an edge, curvature, relative position on the vertical wind profile distribution, and direction of orientation. These features are calculated for every vertex $\mathbf{V}$, along with its pressure, to be used as a sample. The $\mathbf{R}^{23}$ definition of the model is now:

$$f^{ANN} : (Z, \mathbf{n}_{x,y,z}, \mathbf{n}\sigma^{1-5}_{x,y,z}, \mathbf{T}_{x,y,z}) \rightarrow P \tag{2}$$

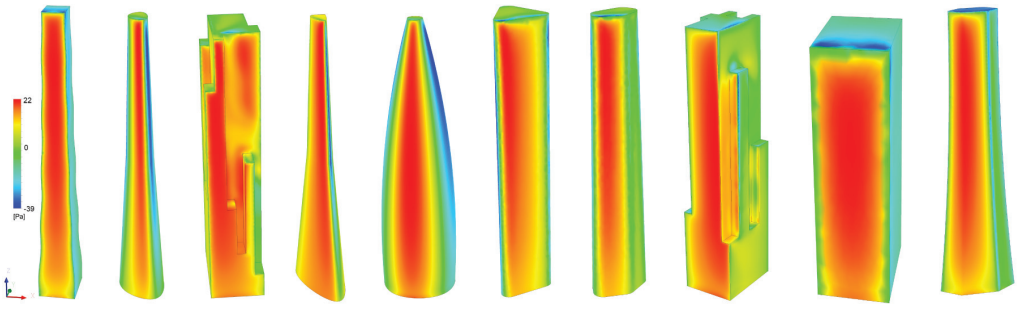
- HEIGHT: Z is the vertical position of $\mathbf{V}$, i.e. $\mathbf{V}_z$;
- NORMAL: $\mathbf{n}_{x,y,z}$ are the normal components of $\mathbf{V}$;
- CURVATURE: $\mathbf{n}\sigma^{1-5}_{x,y,z}$ is the standard deviation of the vertex normals in each independent ring, inversely weighted by the distance. This is also visualised in Figure 3.3.

$$\mathbf{n}\sigma^{r} = \left(\frac{1}{n-1} \sum_{i=1}^{n} \frac{(\mathbf{n}_i - \bar{\mathbf{n}})^2}{d} \right)^{\frac{1}{2}} \tag{3}$$

Where: r is the vertex neighbourhood ring; n is the number of vertices in r; d is the distance between each vertex in n and the central feature vertex; $\bar{\mathbf{n}}$ is the average of the normals in r; $\mathbf{n}_i$ are all the normals in each neighbourhood ring, r.

In this case, convex mesh curvature is given as a positive standard deviation and negative for concave regions. Figure 3.3 shows how this is calculated in 3-D. The basic procedure is as follows:

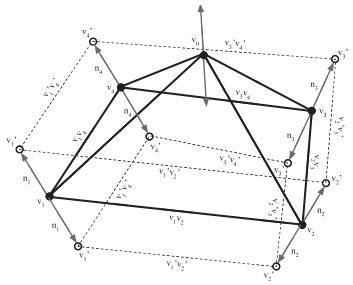
▼ Figure 3.2: Example set of evaluated procedural models.



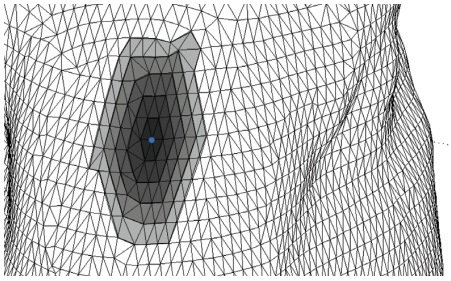
for vertex v_i, its offset is the vertex plus normal $v'_i = v_i + n_i$
if $(v'_1 ... v'_n > v_1 ... v_n)$ then $v_1 ... v_n$ is convex, σ^+
if $(v'_1 ... v'_n < v_1 ... v_n)$ then $v_1 ... v_n$ is concave, σ^-
else $v_1 ... v_n$ is planar, $\sigma = 0$

The extension of the standard deviation to be positive for convex vertex neighbourhood and negative for concave vertex neighbourhoods allows for greater accuracy and applicability to a broader set of forms. The use of various neighbourhood scales (rings one through five) gives the local curvature over a range of scales, as can be seen in Figure 3.4. In these images the white regions are concave and black are convex. Due to the complexity of the meshes here though certain points may in fact be convex in one axis and concave in another. For this reason the *x*, *y*, and *z* components are given for each scale. The images show the mean of all three components to give the primary indication of curvature direction.

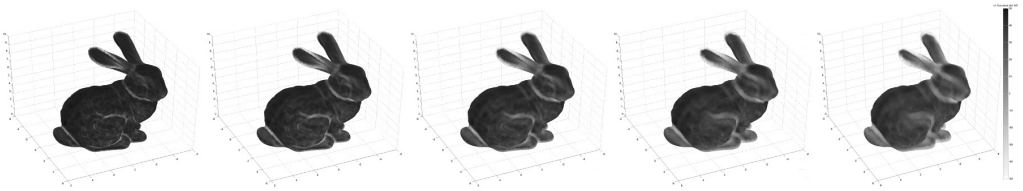
The convex-concave calculation is shown in Figure 3.4 on the Stanford Bunny [30], a standard test model used in computer graphics. The model consists of 69451 triangular polygons, and shows the extension of the curvature neighbourhood from the vertices' first to fifth neighbourhood rings.



(a) $N1$ convexity / concavity calculation.



(b) Vertex neighbourhoods $N1 - 5$.



(a) $N1$-ring (b) $N2$-ring (c) $N3$-ring (d) $N4$-ring (e) $N5$-ring

▲ Figure 3.4: Convex-concave curvature analysis over N1 to N5 neighbourhoods.

- POSITION: $\mathbf{T}_{x,y,z}$ is the normalised position of $\mathbf{V}_i$ within the range of all model vertices $\mathbf{V}$

$$\mathbf{T}_i = \frac{(\mathbf{V}_i - \mathbf{V}_{min})}{(\mathbf{V}_{max} - \mathbf{V}_{min})} \tag{4}$$

- PRESSURE: *P* is simply the pressure at $\mathbf{V}$ as extracted from the simulation. This can quite easily be replaced with any dependent secondary metric, such as force or the pressure coefficient.

The feature generation time, for both training and test models, is currently 0.02784 *s/sample*. For example, a model with a mesh of 1000 vertices currently requires 27.84s. The following pseudocode is a

```
for each mesh {
    read each v{x,y,z} {
        calculate minRange{x,y,z}
        calculate maxRange{x,y,z}
        }
    read each i{a,b,c}
    read each n{x,y,z}
    read each P
    for each vertex {
        for each neighbourhood ring (0 to 5) {
            find indices of connected vertices, e.g. for vertex index a: r0{a}, r1{b,c,d,e,f}, etc.
            }
        for each neighbourhood ring (1 to 5) {
            calculate standard deviation of vertex normals in r1-5
            }
        print vertex feature X{z,n,nσ^1-5,T} and Y{P}
        }
    }
```

The output of the calculation, per vertex, is simply a 23-dimensional vector. E.g. z{0.52}, **n**{-0.68,0.72,-0.05}, $\mathbf{n}\sigma^1$ {-0.03,-0.03,-0.02}, $\mathbf{n}\sigma^2$ {-0.07,-0.07,-0.03}, $\mathbf{n}\sigma^3$ {-0.11,-0.10,-0.02}, $\mathbf{n}\sigma^4$ {-0.15,-0.14,-0.02}, $\mathbf{n}\sigma^5$ {- 0.20,-0.18,-0.03}, **T**{0.79,0.13,0.04}, *P*{-0.17}

From the training feature set, the reduced-order model is generated by a back-propagation artificial neural network (ANN), with a hyperbolic tangent sigmoid transfer function [31]:

$$tansig(x) = 2/(1 + exp(-2 \cdot x)) - 1 \tag{5}$$

The ANN structure *X*:*H*:*Y* is 22:20:1, i.e. 22 input neurons, 20 hidden layer neurons, and 1 output. The sensitivity analysis on the number of neurons in the hidden layer, and the number of layers, is not included here; although 20 in a single layer has been seen to be sufficient. There is generally no rule-of-thumb or guidance to define either, necessitating sensitivity analysis for each problem.

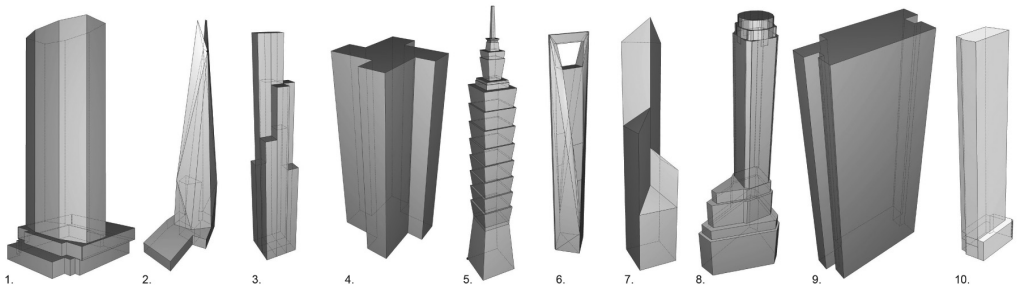
3.4. Test data set

The models are selected from *Google Earth*, rebuilt in *GC* as solids, and evaluated with *CFX*. They were selected relatively arbitrarily, except for the one criteria that each has a unique architectural design feature. For instance, the set contains features such as pedestals (1 and 2), tapering (2), stepping (3 and 7), concavities, corner filleting (5), voids (6), spires (5), etc.

Note that the heights range between 124 and 508*m*, whilst the procedural model used to generate the training set has a height range of 100 to 200*m*. The test models were therefore all scaled to 100*m* to reduce the amount of sampling required for the training set.

◀ Table 3.2: Real building test set details.

	Name	Location	Height $[m)]$	Completion date
1	Met Life Building	New York City, US	246.3	1963
2	The Shard	London, UK	306.0	2013
3	Willis Tower (Sears)	Chicago, US	442.1	1974
4	Euston Tower	London, UK	124.0	1970
5	Taipei 101	Taipei, Taiwan	508.0	2004
6	Shanghai World Financial Centre	Shanghai, China	492.0	2008
7	Bank of China Tower	Hong Kong, China	367.4	1990
8	20 Exchange Place	New York City, US	225.9	1931
9	Frankfurter Buro Center	Frankfurt, Germany	142.4	1980
10	123 Washington Street	New York City, US	192.1	2010



▲ Figure 3.5: Real building test set models.

4. RESULTS

Firstly, sensitivity analyses are conducted on the feature vector components and the training set size using data only from the procedural models; followed by an assessment of the accuracy of the predictions on the real building test set.

The distinction is drawn between the simulation output response *Y* from CFD, and the prediction output response *Y′* from the reduced-order model. For a single vertex sample *i*, the difference between the *Y* and *Y′* is used to calculate the sample prediction error, δ_i :

$$\delta_i = (Y'_i - Y_i)/(Y_{max.} - Y_{min.}) \quad (6)$$

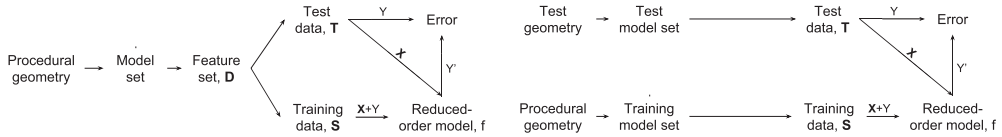
The descriptive statistics used for reporting the errors throughout are:

$\delta_{min.}$	*real-valued minimum of the error range [%]*
$\delta_{max.}$	*real-valued maximum of the error range [%]*
$\overline{\lvert\delta\rvert}$	*mean of the absolute error range [%]* ($\overline{\lvert\delta\rvert} \neq \lvert\bar{\delta}\rvert$)
$\sigma_{\lvert\delta\rvert}$	*standard deviation of the absolute error range [%]* ($\sigma_{\lvert\delta\rvert} \neq \sigma_{\delta}$)

There are two types of test used here: sample-based (Figure 4.1a) or model-based (Figure 4.1b). In the sample-based assessment, test data set **T** of size *m* and training data set **S** of size *n* are drawn from the same set of available data *D*, meaning that $m = D - n$. Both **T** and **S** are randomised in this case, and are used to monitor error convergence during the ANN training. For model-based tests, a completely different test set is used so that **T** and **S** are independently generated, such as in the case where a procedural model is used for training and real models for testing.

Whilst the descriptive statistics for the sample- and model-based accuracy are a good indicator of the ROM's performance, a qualitative visual comparison of simulated and predicted surface pressure is also included at the end.

▼ Figure 4.1: Testing of sample- and model-based data sets.

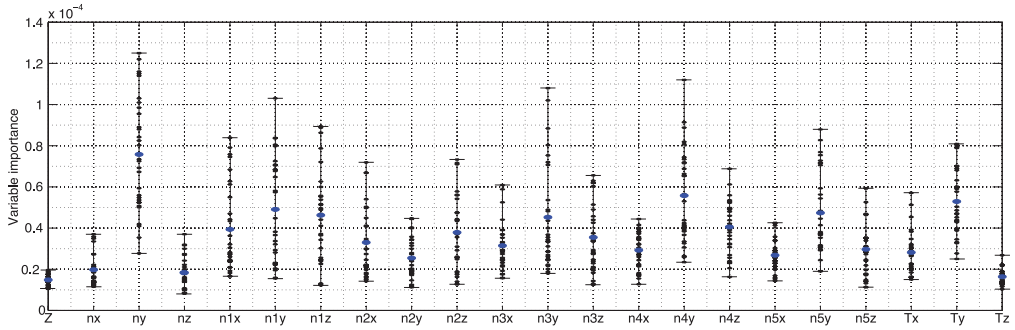


(a) Sample-based (b) Model-based

4.1. Feature sensitivity analysis

Given the feature vector definition of $\mathbf{X}\{z,\mathbf{n},\mathbf{n}\sigma^{1-5},\mathbf{T}\}$, each of the 22 input components has a different significance, importance, or sensitivity to the output. Whilst this varies between problems and geometry, a measure of importance can be calculated during generation of the reduced-order model based solely on the training data set. The random forest method [32], specifically the *TreeBagger* [33] algorithm, intrinsically calculates the *OOBPermutedVarDeltaError*. A set size of 10000 randomly sampled from the full training set is used; 10 trees for the *TreeBagger* algorithm; and the process is re-run 30 times to take the mean and range.

The first observation is that primarily the *y*, and secondarily the *x*, components are typically the most significant part of vector. The *x* (across flow) and the *y* (stream-wise) direction components determine whether the point is facing into, perpendicular to, or away from the flow. Which in turn is the primary indicator of a positive or negative pressure. Also, note that the variability or distribution increases with feature importance: the standard deviation σ and the mean have an r^2 =0.795.



▲ Figure 4.2: Feature importance for $\mathbf{X}\{z,\mathbf{n},\mathbf{n}\sigma^{1-5},\mathbf{T}\}$.

4.2. Training set size sensitivity analysis

Sample-based errors given are at the converged training set size (Figure 4.3) of n=10000. The test set size m, being the full data set D minus the training set n, is therefore 5726831 − 10000 = 5716831. Where the full data set D has $5.727e^6$ samples; an average of 9545 vertices per training model. These are randomly selected for each training run, which is repeated 20 times. The individual runs are shown as grey crosses, with the black lines showing the limits and the blue line the mean over the 20 re-runs. The training set size is increased incrementally, by an increment of 100 from 100 to 1000, and an increment of 500 from 1000 to 10000. The converged sample-based errors are: δ_{min} = -59.365%, δ_{max} = 63.634%, $|\overline{\delta}|$ = 2.767%, and $\sigma_{|\delta|}$ = 3.420%.

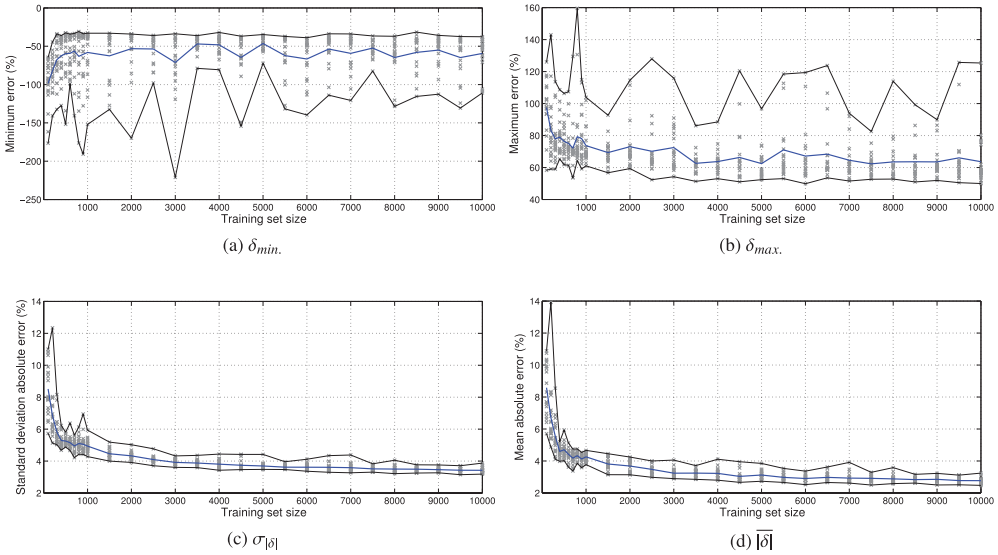
4.3. Model-based test

The test set is of mixed geometric complexity which can be seen in the number of vertices on each model, m, ranging from 528 to 29091. This is a product of the geometric complexity, but, without resampling, also effects the output resolution and speed. Therefore, one of the primary methods for the user to improve the on-line prediction time is to lower the number of vertices on the test model.

A noticeable trend in these 10 test cases is of under-prediction of negative pressure values, especially in localised regions of very low pressure. To be clear, an under-prediction of a negative value means the prediction is too high, e.g. simulated value is -10 but predicted value is -5.

5. DISCUSSION

These developments represent an alternative approach that is fundamentally different to previous attempts at generalising tall building aerodynamics found in the literature. The use of local features rather than global parameters allows for arbitrary complexity in the model and for vertex surface pressure visualisation rather than global factors.



(a) $\delta_{min.}$ (b) $\delta_{max.}$

(c) $\sigma_{|\delta|}$ (d) $\overline{|\delta|}$

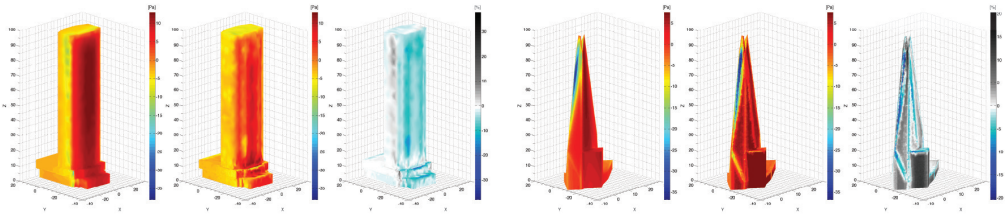
▲ Figure 4.3: Complex geometry: sample-based training error convergence.

Compared to solver approximation techniques, such as the FFD solver, solution approximation has the benefit of being based on a conventional, higher accuracy CFD solver. As such, the validity of the basis data can, to a larger extent, be trusted or verified. The comparative disadvantage is that the FFD can produce field rather than surface data which is useful for identifying flow patterns, assessing pedestrian comfort, and to gauge the secondary downstream effects that a new building will have on others.

A sensitivity analysis was run on the training set size, and found that a sample size of n=10000 was adequate to reach error convergence during the ANN training. The training set is therefore only 0.175% of the full available data set D=5726831 from the 600 evaluated training models.

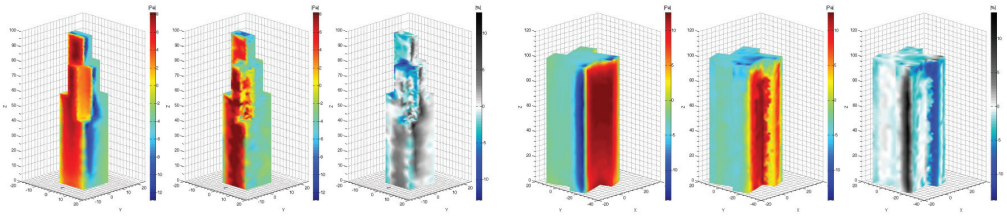
▶ Table 4.1: Error and time results summary - complex geometry: model-based.

			Error, δ [%]				**Test time, t [s]**		
	Case	m	$\delta_{min.}$	$\delta_{max.}$	$\overline{\lvert\delta\rvert}$	$\sigma_{\lvert\delta\rvert}$	X	Y'	$X+Y'$
1	Met Life	11439	-25.018	33.058	3.629	5.088	318.462	0.0468	318.509
2	Shard	20145	-21.899	13.849	2.392	2.498	560.837	0.0680	560.905
3	Sears	1629	-16.025	9.662	2.467	2.838	45.351	0.0230	45.374
4	Euston	1981	-19.770	12.314	3.669	4.059	55.151	0.0238	55.175
5	Taipei101	29091	-17.352	15.027	2.492	2.871	809.893	0.0898	809.983
6	Shanghai	10469	-23.578	25.268	4.440	4.840	291.457	0.0445	291.501
7	BankOfChina	528	-13.702	4.929	2.237	2.479	14.700	0.0203	14.720
8	Exchange	3931	-23.616	23.699	3.929	4.451	109.439	0.0286	109.468
9	Frankfurter	3688	-16.617	13.299	1.994	2.096	102.674	0.0280	102.702
10	Washington	1837	-19.099	13.380	2.766	2.225	51.142	0.0235	51.166



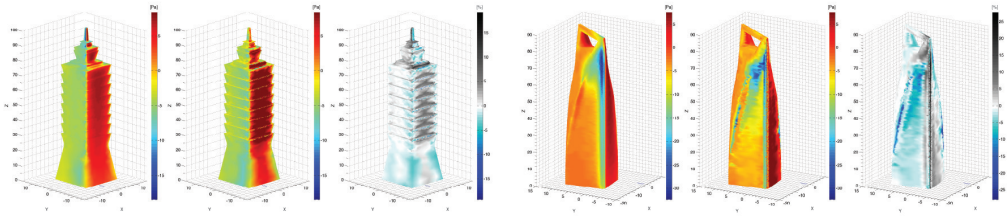
(a) Y (b) Y' (c) δ (d) Y (e) Y' (f) δ

▲ Figure 4.4: Model-based test: (left) Metlife; (right) Shard.



(a) Y (b) Y' (c) δ (d) Y (e) Y' (f) δ

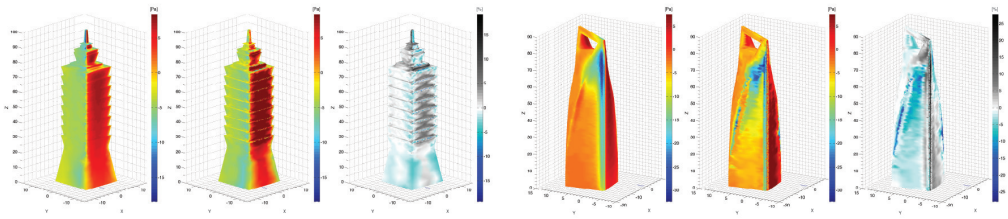
▲ Figure 4.5: Model-based test: (left) Sears; (right) Euston.



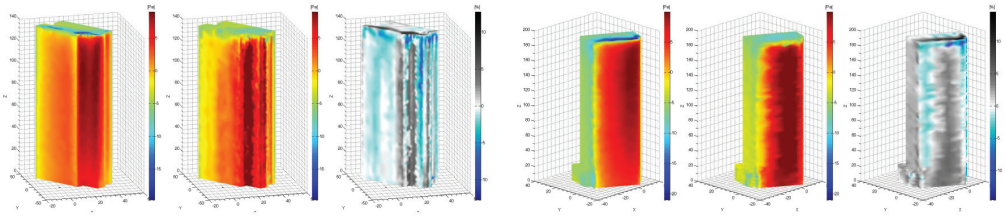
(a) Y (b) Y' (c) δ (d) Y (e) Y' (f) δ

▲ Figure 4.6: Model-based test: (left) Taipei; (right) Shanghai.

▼ Figure 4.7: Model-based test: (left) BankOfChina; (right) Exchange.



(a) Y (b) Y' (c) δ (d) Y (e) Y' (f) δ



▲ Figure 4.8: Model-based test: (left) Frankfurter; (right) Washington.

Although this does not mean that only one training simulation is required (the average number of vertices on a training model was 9545), evaluating 600 models may be excessive. Following this, the final model-based test was visualised to check the predicted pressure distribution qualitatively against the simulation. Generally, under-prediction of negative pressures can be seen, but general patterning or distribution of both positive and negative pressure remains intact.

5.1. Process time analysis

The feature extraction times are based on a calculation speed of 0.02784s=*sample* (about 36 *samples*/s) for off-line ROM generation (n=10000) and on-line predictions. The ANN training time, for n=10000, is averaged over 20 runs; the mean time is 38.269s (σ:17.143s).

The model-based prediction times show that, in comparing only on-line processes, the ROM is 5.39-times faster than the conventional CFD method. However, this does not take into account the full process. By comparing the off-line plus on-line processes for repetition, where x is the number of design iterations, the CFD time=1383.162x and the ROM time=256.482x+1145905.17 (Figure 5.1). In solving for x, the minimum number of iterations before the full ROM process time equals the CFD is x=1017.

5.2. Limitations

A key characteristic of the solution approximation approach is a reduction, where the full field data available through CFD simulation is reduced to the surface data of interest. As such, no direct information about the surrounding wind environment is conveyed as it would be with the full CFD. For certain cases this information is valuable; for instance, pedestrian comfort studies require data on the wind velocity at a horizontal plane above ground level in order to measure the change that a new design can have on an urban environment. The FFD is primarily focused on this field data with surface interactions as a secondary consideration; for the FFD the structured meshes give a poor surface representation and makes direct comparisons with CFD difficult.

Steady-state RANS CFD was used, therefore the prediction results are not time-dependent peaks but time-averaged; a common approximation strategy in practice. This is a relatively significant simplification in structural engineering terms, but necessary to allow simulation of a large training set. In later project stages it is of concern to establish quantifiable peak values for the design, for which more accurate in-depth analyses are conducted, i.e. wind-tunnel or LES.

The algorithm calculating the training and test shape features is not currently optimised for efficiency. The most costly part of the calculation lies in the local curvature analysis, where for every vertex the neighbouring vertices must be found and ordered by proximity. Consequently, the reported tests use a feature generation time of 0.02784*s*=*sample* (about 36 samples per second) which is open for improvement. Similarly for the input feature vector; there is no guarantee that the selection used is necessarily optimal. The local curvature analysis was calculated over five neighbourhood rings per vertex since the computational effort escalates quickly with the number of neighbourhoods. For the shape features, the optimal selection likely varies between geometry types and complexity, i.e. a simpler model would require less features to make an acceptable prediction.

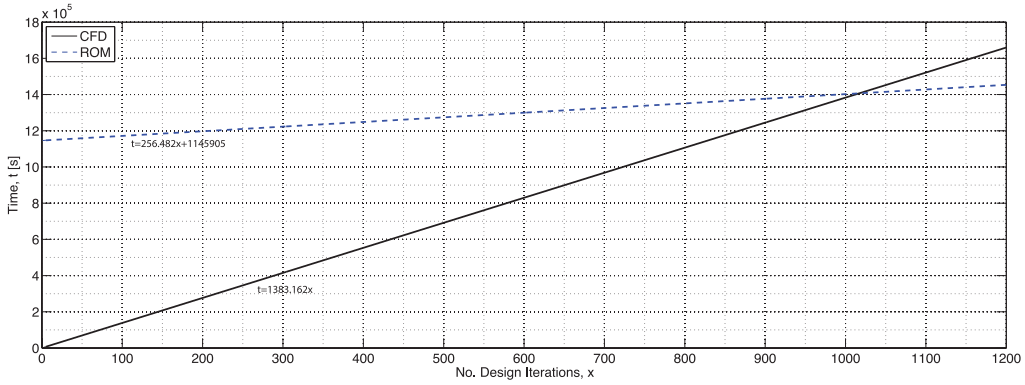
6. CONCLUSION

In summary, the methodology and results presented here demonstrate an alternative approach to approximating tall building wind pressure for generative, early stage design. The results indicate that significant improvements in response time (5.4-times faster when comparing on-line prediction times with conventional CFD) can be made with a reasonable trade-off in accuracy (mean absolute errors of 1.99–4.44% σ:2.10–5.09%). Although the off-line time is substantial, requiring around 1000 predictions before the process time becomes an improvement on the traditional CFD approach, there are three conditions that mediate this limitation: firstly, the

▼ Table 5.1: CFD and ROM process times.

	Time [*s*]			
	Mean	σ	Min.	Max.
Conventional CFD Process				
Test simulation [a]	**1383.162**	1254.994	334.937	4103.020
Off-line ROM Process				
Training simulations [c] (600)	1145905.17	63046.37	941575.2	1220212.36
Feature extraction [b] (n=10000)	278.4	-	-	-
ANN training	38.269	17.143	19.948	62.165
Total	**1146221.84**			
On-line ROM Process				
Feature calculation	256.440	272.215	14.700	809.893
Prediction	0.0414	0.0238	0.0203	0.0898
Total	**256.482**			

[a] Test simulation t=0.132m+265.283 (r^2=0.786); [b] Feature calculation speed = 0.02784s/*sample*; [c] Training simulation individual mean t = 1914.470s (σ:629.808s, range:1034.209–3759.655s).



▲ Figure 5.1: Process time *t* against number of design iterations x for CFD and ROM.

current feature calculation time is not optimised and an improvement on the current speed can easily be attained through more efficient code; secondly, the converged training set sample size of 10000 suggests that 600 training simulations is in fact too high; and thirdly, the off-line process time is inherently a one-off component as compared to the potential infinite number of on-line predictions it enables.

The most promising aspect of the approach is its applicability to higher-order basis CFD. The basic invariance of the on-line prediction time and accuracy to the basis simulation means that benefits increase with cost of the conventional simulation.

ACKNOWLEDGEMENTS

This research was sponsored by the EPSRC, Bentley Systems and PLP Architects.

REFERENCES

1. Stam, J., 1999. Stable Fluids. Tech. rep., Alias-Wavefront, Seattle.
2. Chronis, A., Tsigkari, M., Davis, A. and Aish, F., 2012. Design Systems, Ecology and Time. In *Proceedings of ACADIA12: Synthetic Digital Ecologies*. San Francisco, CA.
3. Chronis, A., Turner, A. and Tsigkari, M., 2011. Generative Fluid Dynamics: Integration of Fast Fluid Dynamics and Genetic Algorithms for Wind Loading Optimization of a Free Form Surface. In *Proceedings of SimAUD SCS SpringSim'11*, 79–86.
4. Karagkouni, C.S., Fatah, A., Tsigkari, M. and Chronis, A., 2013. Fac¸ade Apertures Optimization :Integrating Cross-Ventilation Performance Analysis in Fluid Dynamics Simulation. In *Proceedings of SimAUD SCS SpringSim'13*, Malkawi 2004.
5. Athanailidi, P., Fatah gen Schieck, A., Tenu, V. and Chronis, A., 2014. Tensegrity Systems Acting as Windbreak: Form Finding and Fast Fluid Dynamics Analysis to

Address Wind Funnel Effect. In *Proceedings of SimAUD SCS SpringSim'14*, 127–134. Tampa, Fl.

6. Zuo, W. and Chen, Q., 2009. Real-Time or Faster-than-Real-Time Simulation of Airflow in Buildings. *Indoor Air*, 19(1), 33–44.
7. Zuo, W. and Chen, Q., 2010. Fast and Informative Flow Simulations in a Building by using Fast Fluid Dynamics Model on Graphics Processing Unit. *Building and Environment*, 45(3), 747–757.
8. Graening, L. and Sendho_, B., 2014. Shape Mining: A Holistic Data Mining Approach for Engineering Design. *Advanced Engineering Informatics*.
9. Graening, L., Menzel, S., Hasenj¨ager, M., Bihrer, T., Olhofer, M. and Sendho_, B., 2008. Knowledge Extraction from Aerodynamic Design Data and its Application to 3D Turbine Blade Geometries. *Journal of Mathematical Modelling and Algorithms*, 7(4), 329–350.
10. Degroote, J., Vierendeels, J. and Willcox, K., 2010. Interpolation Among Reduced-Order Matrices to Obtain Parameterized Models for Design, Optimization and Probabilistic Analysis. *International Journal for Numerical Methods in Fluids*, 2010(63), 207–230.
11. Bui-Thanh, T., Willcox, K. and Ghattas, O., 2008. Parametric Reduced-Order Models for Probabilistic Analysis of Unsteady Aerodynamic Applications. *AIAA*, 46(10).
12. Schilders, W., 2008. Introduction to Model Order Reduction. In *Model Order Reduction: Theory, Research Aspects and Applications*. Springer, 13 edn.
13. Rendall, T.C.S. and Allen, C.B., 2008. Multi-Dimensional Aircraft Surface Pressure Interpolation using Radial Basis Functions. *Proceedings of the Institution of Mechanical Engineers*, Part G: Journal of Aerospace Engineering, 222(4), 483–495.
14. Srinivasan, R.S. and Malkawi, A.M., 2004. The Use of Learning Algorithms for Real-Time Immersive Data Visualisation in Buildings. In *SIGRADI*, 329–332.
15. Cipriano, G., Phillips, G.N. and Gleicher, M., 2009. Multi-Scale Surface Descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 15(6), 1201–8.
16. Feng, X., Xia, K., Chen, Z., Tong, Y. and Wei, G.W., 2013. Multiscale geometric modeling of macromolecules II: Lagrangian representation. *Journal of computational chemistry*, 34(24), 2100–20.
17. Darom, T. and Keller, Y., 2012. Scale-Invariant Features for 3-D Mesh Models. *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, 21(5), 2758–69.
18. Jiao, X. and Heath, M.T., 2002. Feature Detection for Surface Meshes. In *Proceedings of 8th international conference on numerical grid generation in computational field simulations*. University of Illinois.
19. Hubeli, A., Meyer, K. and Gross, M., 2000. Mesh Edge Detection. Tech. rep., Swiss Federal Institute of Technology, Zurich.
20. Hubeli, A. and Gross, M., 2001. Multiresolution Feature Extraction for Unstructured Meshes. In *Proceedings of the conference on Visualization'01*. *IEEE Computer Society*, 287–294.
21. Szilvasi-Nagy, M., 2006. About Curvatures on Triangle Meshes. In *KoG 10 - Journal of Croatian Society for Geometry and Graphics*, 13–18.
22. Dong, C.S. and Wang, G.Z., 2005. Curvatures Estimation on Triangular Mesh. *Journal of Zhejiang University SCIENCE*, 6(Suppl. I), 128–136.
23. Walter, N., Laligant, O. and Aubreton, O., 2008. Salient Point SUSAN 3D Operator for Triangles Meshes. In *The 2nd International Topical Meeting on Optical Sensing and Artificial Vision*.

24. Bentley Systems, 2013. GenerativeComponents v08.11.08.296. www.bentley.com.
25. Park, S.M., Elnimeiri, M., Sharpe, D.C. and Krawczyk, R.J., 2004. Tall Building Form Generation by Parametric Design Process. In *CTBUH*, 1–7. Seoul Conference.
26. Samareh, J.A., 2001. Survey of Shape Parameterization Techniques for High-Fidelity Multidisciplinary Shape Optimization. *American Institute of Aeronautics and Astronautics*, 39(5), 333–343.
27. ANSYS, 2014. CFX v13.0. www.ansys.com.
28. ANSYS, 2009. CFX-Solver Theory Guide. Tech. rep., ANSYS Inc., Canonsburg, PA.
29. Hsu, S.A., Meindl, E.A. and Gilhousen, D.B., 1994. Determining the Power-law Wind-Profile Exponent under Near-Neutral Stability Conditions at Sea. *Journal of Applied Meteorology*, 33(1994), 757–772.
30. Turk, G. and Levoy, M., 1994. The Stanford 3D Scanning Repository. http://graphics.stanford.edu/data/3Dscanrep/.
31. Vogl, T.P., Mangis, J.K., Rigler, A.K., Zink, W.T. and Alkon, D.L., 1988. Accelerating the Convergence of the Back-propagation Method. *Biological Cybernetics*, 59(4-5), 257–263.
32. Breiman, L., 2001. Random Forests. *Machine Learning*, 2001(45), 5–32.
33. Matlab. TreeBagger algorithm. http://www.mathworks.co.uk/help/toolbox/stats/treebaggerclass.html

Samuel Wilkinson and Sean Hanna

University College London
Bartlett School of Graduate Studies
132 Hampstead Road

S. Wilkinson, ucftsmw@ucl.ac.uk

A.3 Wilkinson, S. and Hanna, S. (2014). Reduced-Order Topological Performance Models. In *proceedings of IBPSA-CIBSE BSO14: Building Simulation and Optimisation*.

REDUCED-ORDER TOPOLOGICAL PERFORMANCE MODELS

Samuel Wilkinson[1] and Sean Hanna[2]
[1]samuel.wilkinson.09@ucl.ac.uk, [2]s.hanna@ucl.ac.uk

University College London
Bartlett School of Graduate Studies
14 Upper Woburn Place
London, WC1H 0NN

ABSTRACT

In this paper an approach for generating reduced-order performance models from surface mesh topology is presented. The method uses an artificial neural network (ANN) to create a regression function linking local vertex shape characteristics and simulation response. Two cases of model orientation interpolation are demonstrated; firstly for simple insolation; and secondly for wind pressure from steady-state computational fluid dynamics (CFD) simulations. Finally, both are integrated in a single- and multi-objective analysis of the performance space and prediction variability, and an assessment of the approach's speed and accuracy. It is concluded that, since prediction time is independent of the basis simulation, the benefits increase with simulation cost; and that prediction variability, or error, does not substantially alter the structure of non-dominated solutions in the Pareto analysis.

INTRODUCTION

Model reduction refers to the generation of approximative representations of system behaviours, primarily for time-consuming computational simulations. The aim is fundamentally to create a lower-dimensional system model of the inputs and outputs, whilst retaining predictive fidelity (Bui-Thanh et al., 2008; Schilders, 2008). Analyses that are potentially obstructive to the design process may involve partial differential equations (PDEs), such as the Navier-Stokes equations of fluid flow and the Maxwell equations in electromagnetics (Degroote et al., 2010).

Reduced-order models (ROM) are derived from such high-fidelity simulations or models and restrict the output quantities to those of interest (e.g. lift or drag, or a quantity at a single point). Model order reduction originated in systems and control theory fields where there was a strong need to reduce the complexity of dynamic systems in order to study them. These systems may have of the order of 10^5 to 10^9 equations or variables, and yet their input-output behaviour must be preserved.

Architectural generative design practice is trending towards integration with environmental performance data to guide iterative form exploration and optimisation (Malkawi, 2004). However the aforementioned simulations, like computational fluid dynamics (CFD) for fluid flow, constrain this process. Therefore reduced-order models can be used for lightweight, approximate predictions at early design stages when guidance is valuable for assessing multitudes of alternatives and for optimisation (Robinson et al., 2008).

The need for application-specific simulation accuracy and speed that meets the demands of early design stages is proposed by Lu et al. (1991); a range of reduced-order models of a combustion engine simulation are generated, with varying accuracies and speed that can be used throughout the design process. The solution is posed as a Pareto front of non-dominated solutions, rather than a simpler trade-off curve based on biological decision making (Chittka et al., 2009) as otherwise suggested (Aish et al., 2012).

Optimisation of early design stage building form, in this case topology, with regards to aerodynamics (Kim et al., 2011) has been attempted with a genetic algorithm and CFD. The output was limited by the speed of the evaluation stage and the large number of simulations required for stochastic optimisation. Whilst efforts towards parallelisation assist in optimisation problems (Mueller and Strobbe, 2013), it offers no assistance to individual evaluation time.

Reduced-order models can be a potential solution to this issue. Generated by machine learning algorithms, such as artificial neural networks (ANNs), training data sets of evaluated models effectively sample the design space and are used to create a regression function between input feature vector and typically a single-dimensional system response. The regression function itself constitutes the reduced-order model representing a limited region of interest of the underlying physical system.

Conventionally in existing studies, especially on wind interference (Khanduri et al., 1998), global model parameters are used for the input feature vector (such as position, orientation, width, height, etc.). However this approach has two limitations: firstly, a top-down user description of the design often constrains representation; and secondly, a global input implies a global output, offering limited resolution.

The approach described here solves both of these issues by using local vertex descriptors for the input vector along with a corresponding performance metric for that vertex. In this way each vertex, or sample point on the model's surface, is treated independently from the global simulated domain. The core premise of this approach to reduced-order modelling is that this domain decomposition is viable whilst maintaining the integrity of the system's behaviour.

The use of triangulated surface meshes is well-suited here since it offers a pre-sampled representation of the original model and simple connectivity information. Triangulated meshes are also robust (full volumetric closure is not required), ubiquitous in practice (a mesh may already exist from rendering or other analysis), and is not associated with any specific type of geometry or simulation. This allows great flexibility and many opportunities both upstream and downstream from the model reduction process.

Graening et al. (2008) use a similar mesh topology to predict the pressure sensitivity against deformation displacement of individual vertices on a turbine blade CFD mesh. A decision tree is used to create a reduced-order model of this relatively constrained aspect of an otherwise complex problem, thereby extracting useful knowledge for subsequent designs.

In fact, mesh topology and local shape feature calculation is used in a broad range of fields such as image stabilisation, geometric reconstruction, or even biochemistry. Cipriano et al. (2009) use local shape features to analyse surface regions to identify complementary molecule shapes which is related to chemical functionality. In this case, curvature and degree of anisotropy (directionality) are used.

The method has been demonstrated on specific cases of wind-induced surface pressure derived from CFD on cuboid orientation, height, and topological interpolation, and realistic, context-free tall building prediction from procedural models (Wilkinson et al., 2013). In this case the reduced-order model is generated and tested on different geometries with a similar prediction time and accuracy as this study. As well as for cases of complex urban wind interference where it has been demonstrated how a context-free prediction can essentially be super-imposed onto a contextual wind simulation through a similar method used here (Wilkinson et al., 2014).

In this paper, the methodology outlined in these two previous studies is generalised and extended to integrate CFD and insolation towards a multi-objective optimisation problem. Wind pressure P $[Pa]$ and insolation I $[W.h]$ are incommensurable metrics, i.e. they are not directly comparable and one cannot necessarily be designated as more significant than the other. Therefore unless weights are assigned, a Pareto-efficient set of non-dominated solutions can be created which essentially defers this designation to a higher level of user decision. The clear advantage however is that all of the alternatives in the Pareto set are optimal with regards to one objective or another.

METHODOLOGY

A description of the mesh definition, reduced-order model creation, and simulation methodology are presented in this section.

Mesh Representation

The process of meshing can be seen as discretising or sampling the true model which may originally consist of complex surfaces or solids. Whilst there are many different types of polygon meshes, the most common and simplest is the triangulated. The original geometric model can be represented as an unstructured triangulated surface mesh consisting of a list of vertices, $\mathbf{v}$, normals $\mathbf{n}$, and a list of indices, i, in connectivity triplets defining a face, F.

Here, the positions of $\mathbf{v}$ are essentially sample points of the original surface $\mathbf{v}\{x, y, z\}$; $\mathbf{n}$ gives the normals perpendicular to the surface at $\mathbf{v}$, $\mathbf{n}\{x, y, z\}$; and each triangular planar face is defined as $\mathbf{F}\{i_1, i_2, i_3\}$ corresponding to the vertex indices. This basic mesh definition is shown is Figure 1.

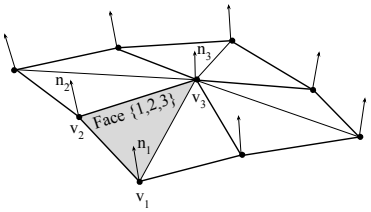


Figure 1: *Triangulated surface mesh format.*

Feature Definition

In defining the reduced-order model f it takes the generalised form:

$$f : \mathbf{X} \rightarrow Y \tag{1}$$

where $\mathbf{X}$ is the n-dimensional vertex description and Y is the vertex's performance response to be predicted. Here, the input feature vector $\mathbf{X}$ consists of: the height Z; the normal components $\mathbf{n}\{x, y, z\}$; the local curvature over various neighbourhood scales $\mathbf{n}\sigma^{1-5}\{x, y, z\}$; and the normalised position $\mathbf{T}\{x, y, z\}$. The output response Y is either the wind pressure P or the insolation I. The reduced-order model can therefore be summarised as:

$$f : (Z, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{T}) \rightarrow Y \tag{2}$$

The local curvature component is calculated for each independent neighbourhood ring of connected vertices. For each ring, from the first to the fifth, the set of vertex normals are found and the standard deviation of the x, y, and z components is calculated. The focal vertex (v_0) and

its five neighbourhood rings are shown in Figure 3 on a typical triangulated surface mesh. To consider convex-concave curvatures, σ is either a real-valued positive or negative number respectively. The process for determining convexity on the first neighbourhood ring is shown in Figure 2, and in the following procedure:

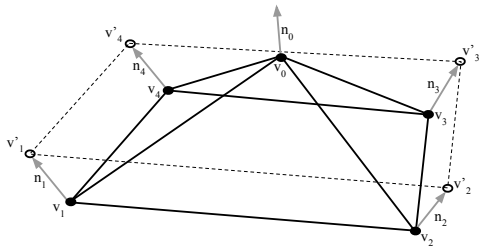


Figure 2: *Local curvature calculation.*

for $\boldsymbol{v}_0$ *ring-1 vertices,* $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4\}$
each is offset by its normal, e.g $\boldsymbol{v}_1 + \boldsymbol{n}_1 = \boldsymbol{v}'_1$
if $(\boldsymbol{v}'_1\boldsymbol{v}'_2\boldsymbol{v}'_3\boldsymbol{v}'_4) > (\boldsymbol{v}_1\boldsymbol{v}_2\boldsymbol{v}_3\boldsymbol{v}_4)$, $\boldsymbol{v}_0$ *is convex* $(+\sigma)$
if $(\boldsymbol{v}'_1\boldsymbol{v}'_2\boldsymbol{v}'_3\boldsymbol{v}'_4) < (\boldsymbol{v}_1\boldsymbol{v}_2\boldsymbol{v}_3\boldsymbol{v}_4)$, $\boldsymbol{v}_0$ *is concave* $(-\sigma)$
else, $\boldsymbol{v}_0$ *is planar* $(\sigma = 0)$

This procedure is essentially measuring the perimeter distance around a ring of vertices, and the distance around the same vertices when offset by their normals. If the offset perimeter is greater the curvature is convex, and vice versa.

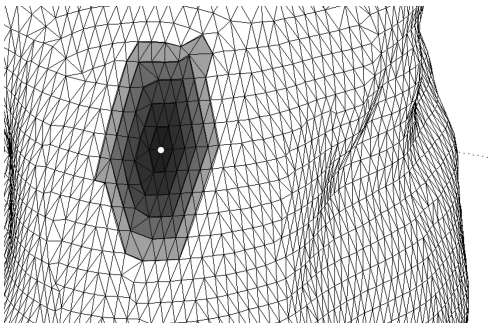


Figure 3: *Vertex neighbourhoods.*

For the training and test meshes, the co-ordinates, normals, and connectivity of a vertex are used to generate the $\mathbf{R}^{22}$ input feature vector sets. Every vertex on each model is used at this stage, however it is shown that not all of the data is required for generating the reduced-order model.

Tall Building Model

The geometry used for testing is an early-design stage model of a tall building (see Figure 4). It was initially generated as a solid (a continuous volumetric object) in *Bentley's GenerativeComponents* v08.11.09.110. The geometry is exported for simulation as a set of surfaces in IGES format, before being meshed. However, the shape analysis can be undertaken directly at this stage by exporting as either PLY, OBJ, or STL, or another mesh file format.

The model has a height of $310m$ (Z-axis), a cross-wind (X-axis) width of $55.4m$, and an along-wind (Y-axis) depth of $51.4m$. The aspect ratio (width:height) is therefore roughly 1:6.

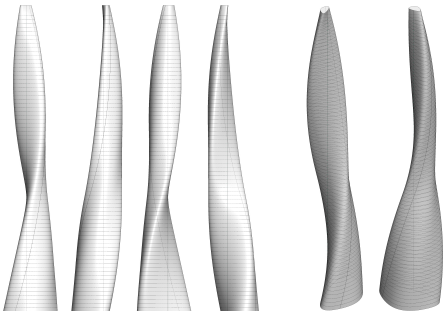


Figure 4: *Design test model.*

Simulation

The wind-induced surface pressure is evaluated through steady-state Reynolds-averaged Navier-Stokes (RANS) computational fluid dynamics (CFD), using *ANSYS CFX* 13.0. The steady-state simulation uses Reynolds decomposition to separate out the fluctuating and average components of the flow. The reported surface pressure is therefore an average of what may potentially be a fluctuating quantity (i.e. not peak values over time), however this is a common approximation in practice especially for the given early stage application. The basic parameters used for the CFD simulations are given in Table 1.

Table 1: *CFD simulation parameters.*

PARAMETER	DESCRIPTION
Turbulence model / Wall function	$k-\epsilon$ Scalable
Advection scheme / Turbulence numerics	High resolution
Residual convergence target	$1.00e^{-6}$ RMS
Heat transfer model	Isothermal 25°C
Reference wind speed	$10m/s$
Domain size ($X:Y:Z$)	600:1200:600m
Min. mesh edge size	$0.3m$
Average no. domain elements	185500

A vertical power-law wind profile is applied to the inlet boundary, with a reference speed u_r of $10m/s$, at a reference height z_r of $10m$, and with exponent α of 0.143 for open-country (Hsu et al., 1994):

$$u_x = u_r \cdot (z_x/z_r)^{\alpha} \tag{3}$$

The mesh generated for the CFD was subsequently re-used for the insolation. The insolation is calculated implicitly as a function of the dot product between vertex normal and a given sun position. This basically gives an intensity based solely on the vertex's normal. Therefore the values of I [-] given subsequently are a proxy of the

real insolation calculation [*W.h*]. For example, in Bergen, Norway, the peak sun altitude at noon on December 21st is 6.3° (tan(6.3)). Therefore, assuming north is the +Y-axis the solar vector is [0,1,-0.1104]. Both the insolation and pressure for the 0° model are shown in Figure 5.

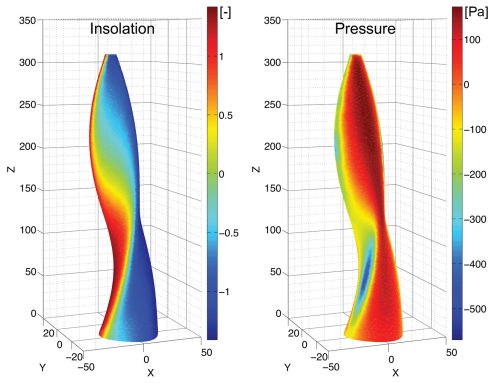


Figure 5: *Simulations of 0° orientation: (left) insolation [-]; (right) wind pressure [Pa].*

Machine Learning

For all cases, the training sets are generated from discrete orientation, O, intervals of 90° (0, 90, 180, 270°). The remainder of the orientations, at the intermediate 15° intervals (15-75, 105-165, 195-255, 285-345°), are used for testing. The two basic experiments are therefore the rotational interpolation of a model's insolation and pressure.

From the training feature sets, the reduced-order model is generated by a back-propagation artificial neural network (ANN), with a hyperbolic tangent sigmoid transfer function (Vogl et al., 1988):

$$tansig(x) = 2/(1 + exp(-2 \cdot x)) - 1 \qquad (4)$$

The ANN structure, shown notionally in Figure 6, was typically 22:20:1, i.e. 22 input neurons, 20 hidden layer neurons, and 1 output. A sensitivity analysis on the hidden layer size is run for both cases.

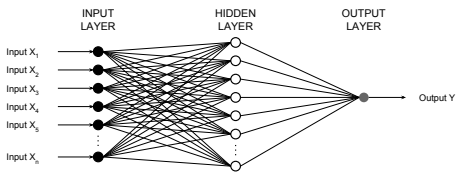


Figure 6: *ANN structure, X:H:Y, 22:20:1.*

Pareto Analysis

The aim of the single- and multi-objective Pareto analysis in the following section is to show that: i) the performance space is broad and complex even for a trivial problem such as orientation interpolation; and ii) the prediction errors are relatively small when compared with the entire performance space. The magnitude of the errors will determine whether or not the predictions alters the structure of the non-dominated solutions and Pareto fronts in the insolation-pressure domain.

RESULTS

The results are presented in three parts: i) insolation vs. orientation interpolation; ii) pressure vs. orientation interpolation; and iii) selective single- and multi-objective analysis to integrate both performance aspects.

The mean absolute error δ for an individual test model m, i.e. the model-based error:

$$\delta_m = (\Sigma(|Y_i' - Y_i|)/m)/range(Y_M) \cdot 100 \qquad (5)$$

and for M, the total test set over 20 models, i.e. the sample-based error:

$$\delta_M = (\Sigma(|Y_i' - Y_i|)/M)/range(Y_M) \cdot 100 \qquad (6)$$

where, Y_i' is the individual vertex prediction and Y_i the simulated value The range is defined over the full test set:

$$range(Y_M) = max(Y_M) - min(Y_M) \qquad (7)$$

From the 20 test models, m_{mean}=4682, m_{min}.=4616, m_{max}.=4729, M=93508; and from the 4 training models N=18862.

Insolation

Figure 7 shows the mean absolute error δ_M [%], against the training set size N. The error is calculated as the difference between the predicted Y' and simulated Y results for the entire test set M. The ANN was run 10 times (R=10) for each set size, and the mean error taken, to account for variability. It can be seen that after N=300, δ_M converges to 0.011% with a standard deviation σ of 0.017%.

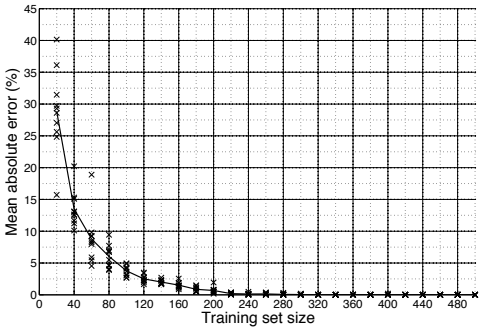


Figure 7: *Training convergence, N against δ_M (R=10, H=20).*

The required number of neurons in the hidden layer is typically a sign of the problem complexity. In this first case, the output (insolation) is essentially a linear product of the input (vertex normals). This is especially true here since the model

is free of over-shadowing. In Figure 8, where the number of hidden layer neurons H is varied between 1 and 20, there is little variation in δ_M. If there is any trend, although negligible, δ_M increases slightly with H, perhaps due to noise introduced by an overly-complicated setup.

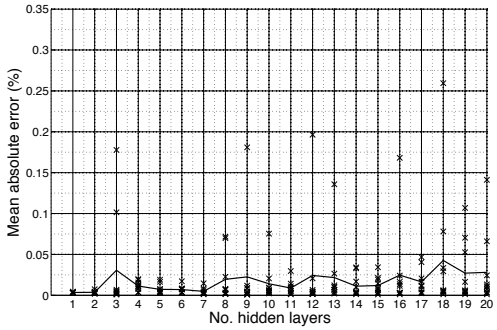


Figure 8: *Hidden layer size sensitivity analysis, H against δ_M (R=10, N=300).*

The variation of mean insolation $\bar{I}$ against orientation O is shown in Figure 9. The blue line is the simulated data, whilst the grey to black lines are for various training sets N of increasing size. The training intervals are shown in bold vertical lines. As N increases, the predictions converge towards the simulated values. By N=160 the difference between prediction and simulation becomes negligible.

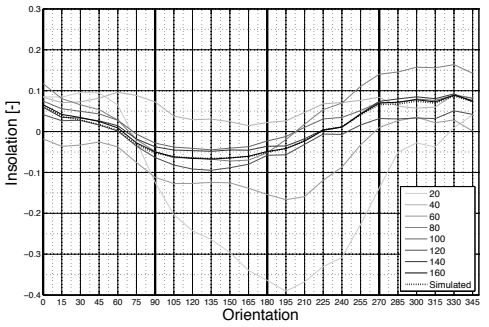


Figure 9: *O against $\bar{I}$ (N varies, H=20).*

Wind Pressure

Now for the wind pressure case, δ_M converges to 5.076% after N=3000 with a standard deviation σ of 0.462% (Figure 10).

There is considerably more variability in the converged model than with the insolation case (σ=0.462% compared to σ=0.017%). Comparing against the insolation case, there is a marked convergence in the ANN prediction error as δ_M decreases with H (Figure 11). After H=20 the error is largely converged, however due to the variability a R >10 and H >20 could be used.

The variation of mean surface pressure $\bar{P}$ against orientation O is shown in Figure 12. The blue

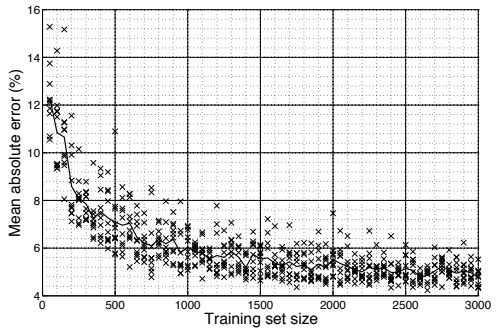


Figure 10: *Training convergence, N against δ_M (R=10, H=20).*

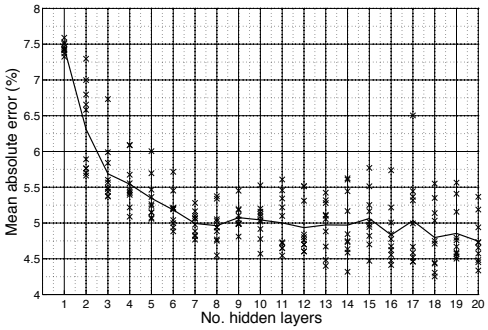


Figure 11: *Hidden layer size sensitivity analysis, H against δ (R=10, N=5000).*

line shows the simulation, and the grey through black increasing values of N. The training intervals are shown in bold vertical lines. Again there is a greater variability than in the previous case. After N=3000, although differences remain greater than with the previous case, the general behaviour of P against O is achieved to an acceptable degree.

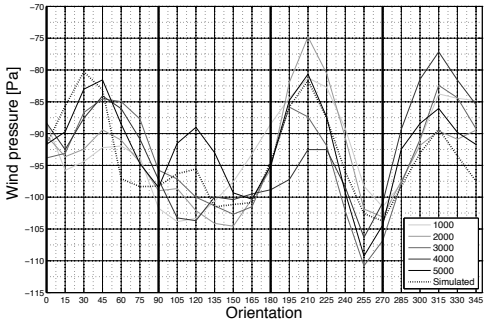


Figure 12: *O against $\bar{P}$ (N varies, H=20).*

In a real design scenario, only the training data would be available to generate the reduced-order model, i.e. simulations at 0, 90, 180, and 270° orientations, meaning it would not be possible to validate the test cases' accuracy. The previous two studies suggest that from these four input simulations, the insolation and wind pressure for

any number of subsequent orientations can be predicted with a reasonable accuracy.

Single-Objective Analysis
For the case of optimisation an objective function must be specified, i.e. minimisation or maximisation of either pressure or insolation. For the single-objective case the solution selection is trivial. From Figure 13 the minimum and maximum $\bar{I}$ and $\bar{P}$ orientations can be seen: $min.(\bar{I})$=135°; $max.(\bar{I})$=330°; $min.(\bar{P})$=255°; $max.(\bar{P})$=210°.

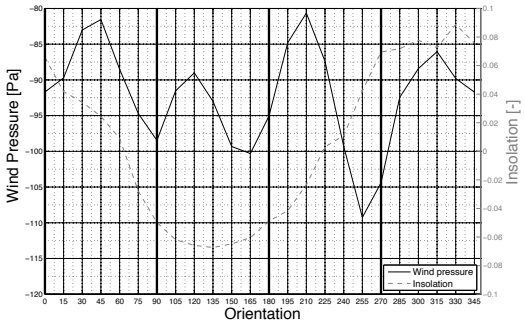


Figure 13: *O against $\bar{P}'$ and $\bar{I}'$.*

Note that orientations for neither the minimum nor maximum of $\bar{I}$ or $\bar{P}$ are included in the original training model set. This implies that the reduced-order model is capable of making test predictions outside of the original training set range, an advantage over simpler linear interpolation.

Although the reduced-order model allows a trivial selection of the optimal design orientation in this interpolation case, a scenario with more complex parameters, e.g. shape topology, would require an optimisation search routine such as a simple gradient descent or stochastic genetic algorithm.

Pareto Analysis
The multi-objective problem differs from the single-objective in that there is typically no single 'best' solution. Instead, the solution set represents various trade-offs between opposing objectives (Veldhuizen and Lamont, 1998). Such problems generally always have objectives that are incommensurable, otherwise the problem would be reducible to a single-objective one and solved finding the best single solution again becomes relatively trivial.

Although the single-objective problem can now be solved easily for either insolation or pressure, the multi-objective analysis remains. The Pareto fronts are shown in Figure 14 where wind pressure is plotted against insolation.

In Figure 14, dominated (internal, ⬧ markers) and non-dominated (boundary, ■ markers) solutions are shown. The two rings connect the non-dominated solutions for the predictions (black) and simulations (blue). The predicted and simulated points are connected with grey lines as a 2-D error bar.

The dominated (⬧) solutions can essentially be discarded at this point since there is a better alternative according to any objective. The non-dominated (■) solutions make up the four Pareto frontiers, dependent on whether $\bar{I}$ or $\bar{P}$ are to be maximised or minimised.

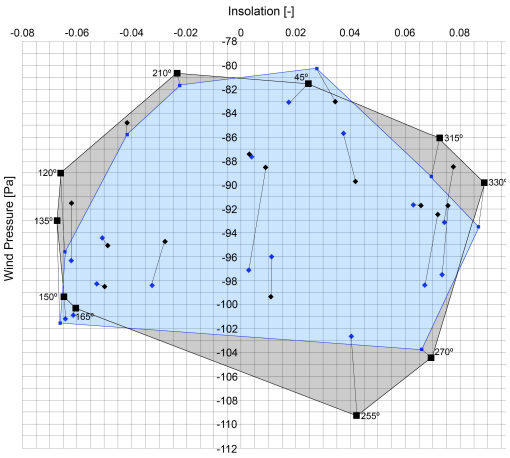


Figure 14: *$\bar{P}$ vs. $\bar{I}$ (simulated: blue) and $\bar{P}'$ vs. $\bar{I}'$ (predicted: black).*

This decision is now for the designer and the specific environmental requirements of the site and building function. To apply the example design conditions mentioned earlier with the sun position (Norway), a notional cold climate can be assumed where pressure is minimised (to avoid infiltration) and insolation maximised (to increase heat gains), i.e. $min(\bar{P}), max(\bar{I})$. For this climate, there are therefore five potential solutions in the Pareto set. Ranging in importance from P to I, the orientations are: 210, 45, 315, and 330°.

DISCUSSION

The primary aims of this work are to: demonstrate the scalability of the reduced-order topological model approach for two different simulation bases (one trivial, insolation; and one costly, wind-induced pressure); show the relative complexity of the P-I performance space even for such a simple problem as model orientation; and to show the reduced-order model prediction errors relative to this performance space.

Figure 14 shows the relative complexity of the P-I performance space. The full set of simulated non-dominated solutions (■) can be compared with the predicted set (■) Considering the predicted set is purely based on the four training models, the structure of the Pareto fronts remains coherent. The primary distortion is in P rather than I: the model-based errors in Table 3 show a mean absolute error of around 14% between $\bar{P}$ and $\bar{P}'$, as compared with 0.3% between $\bar{I}$ and $\bar{I}'$.

Time Versus Accuracy
The aim of generating reduced-order models is to increase the speed at which predictions can be made whilst maintaining accuracy. Table 2 gives the error for the sample-based tests, i.e. training (Figures 7 and 10). And table 3 gives the model-based errors, i.e. the test model errors (Figures 14).

Table 2: *Sample-based prediction errors.*

ERROR	**I**	**P**
Minimum [%]	0.001	4.233
Maximum [%]	0.203	7.459
Mean abs. error, δ_M [%]	0.011	5.076
Standard deviation abs. error, σ [%]	0.017	0.462

Table 3: *Model-based prediction errors.*

ERROR	**I**	**P**
Minimum [%]	0.021	0.059
Maximum [%]	0.580	36.566
Mean abs. error, δ_M [%]	0.327	14.148
Standard deviation abs. error, σ [%]	0.157	11.345

Table 4 compares the process times of the conventional simulation against the presented reduced-order model approach's offline and online components. In making a comparison, the generation of the reduced-order model should be considered in two parts: the offline one-time training stage; and the infinitely repeatable online testing stage.

Table 4: *Model-based prediction times.*

SIMULATION TIME [s]	**I**	**P**
Simulation (/model)	13.920	656

TRAINING TIME [s]	**I**	**P**
Training simulations (4no.)	55.680	2624
Feature calculation (4no.)	521.388	521.388
ANN training	2	21
Total offline	579.068	3166.388

TEST TIME [s]	**I**	**P**
Feature calculation (/model)	130.347	130.347
Prediction (/model)	0.023	0.023
Total online	130.370	130.370

For the insolation case, the conventional simulation time is negligible since the insolation is simply calculated via the dot product. This is only possible because there is no over-shadowing from any surrounding context. It is therefore simply faster to calculate the insolation for each new orientation and the use of a reduced-order model is not warranted. The process time ratio between simulation:prediction (t_Y : $t_{Y'}$) is 13.92:130.37, i.e. it is 9-times slower to use the reduced-order model.

In contrast for the wind pressure case, the ratio is 656:130.37, meaning it is around 5-times faster to use the reduced-order model over the CFD simulation. This difference grows as the CFD simulation time increases with resolution or complexity, meaning the approach becomes more beneficial, i.e. the reduced-order model prediction time is independent of the basis simulation. Therefore, for higher cost CFD such as large eddy simulation (LES) or direct numerical simulation (DNS) the benefits will be even greater.

Generalisation
As well as potential applicability to a range of different analysis methods, the flexibility of the presented methodology has the scope for generalisation both upstream and downstream of the model reduction process. This is a claim made in conjunction with previous work by Wilkinson et al. (2013, 2014), where a reduced-order model has been demonstrated for approximating CFD with an input of procedural models and a test set of real tall buildings.

The prevalence of triangulated surface meshes in design and simulation can be capitalised on with this approach. They offer a pre-sampled representation of the original model, free of any association with a particular simulation type or logic to its creation. If a reduced-order model has been created beforehand offline, only a mesh is required for testing, largely avoiding the expertise required to set up a CFD simulation. This is a significant benefit of this approach to stress: many of the technicalities involved in CFD are embedded in the training process, primarily mesh generation and boundary condition definition. The end-user can therefore simply conceptualise the process as geometric, which as a reduction or simplification is beneficial for generative design.

Downstream of the reduced-order model creation there are also opportunities for the collection and sharing of large sets of performance data, subject to further work. For instance, an online repository of models, features and data can be accumulated and improved over time by users. This can potentially be integrated with a parametric CAD tool that facilitates the interaction between local user, online prediction service, or any subsequent optimisation process.

CONCLUSION

A method has been presented for generating reduced-order models of insolation and wind pressure from local model topology. It has been implemented on a case of interpolating model performance against orientation, and the subsequent analysis of single and multiple objectives on an example design scenario.

It was shown that, as expected, there was no improvement (9-times slower) in prediction time for the insolation case, but that there was considerable improvement (5-times faster) for the wind pressure. The benefits of the reduced-order model are obviously related to the cost or speed of the simulation, i.e. the prediction time is independent of the basis simulation time. Therefore, if self-shading (occlusions) and a recursive ray-tracing

method were used for the insolation then the increased simulation time would warrant model reduction. The Pareto analysis shows that the error variability is relatively small compared with the performance space, and that the errors do not substantially alter the structure of the non-dominated solutions on the Pareto front.

ACKNOWLEDGEMENTS

This work was sponsored by the UK *Engineering and Physical Sciences Research Council*, *Bentley Systems, Inc.*, and *PLP Architecture, Ltd.*.

NOMENCLATURE

$\rightarrow$	Input-output mapping
δ_m	Mean absolute model-based error [%]
δ_M	Mean absolute sample-based error [%]
f	Reduced-order model
H	No. hidden layer neurons
$\bar{I}$	Mean model insolation [-]
m	Model-based test set size
M	Sample-based test set size
$\mathbf{n}$	Vertex normal $\{x, y, z\}$
N	Training set size
O	Orientation [°]
$\bar{P}$	Mean model pressure $[Pa]$
R	No. re-runs
$\mathbf{R}$	Dimensionality
σ	Standard deviation [%]
$\mathbf{T}$	Relative vertex position on model $\{x, y, z\}$
$\mathbf{X}$	Input feature vector $\{Z, \mathbf{n}, \mathbf{n}\sigma^{1-5}, \mathbf{T}\}$
Y	Simulation output
Y'	Prediction output

REFERENCES

Aish, F., Chronis, A., Tsigkari, M., and Davis, A., 2012. Design Systems, Ecology and Time. In *proceedings of ACADIA*.

Bui-Thanh, T., Willcox, K., and Ghattas, O., 2008. Parametric Reduced-Order Models for Probabilistic Analysis of Unsteady Aerodynamic Applications. *American Institute of Aeronautics and Astronautics*, 46(10).

Chittka, L., Skorupski, P., and Raine, N.E., 2009. Speed-Accuracy Tradeoffs in Animal Decision Making. *Trends in Ecology and Evolution*, 24(7), pp.400-407.

Cipriano, G., Phillips, G.N., and Gleicher, M., 2009. Multi-Scale Surface Descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 15(6), pp.1201-8.

Degroote, J., Vierendeels, J., and Willcox, K., 2010. Interpolation Among Reduced-Order Matrices to Obtain Parameterized Models for Design, Optimization and Probabilistic Analysis. *International Journal for Numerical Methods in Fluids*, (63), pp.207-230.

Graening, L., Menzel, S., Hasenjger, M., Bihrer, T., Olhofer, M., and Sendhoff, B., 2008. Knowledge Extraction from Aerodynamic Design Data and its Application to 3D Turbine Blade Geometries. *Journal of Mathematical Modelling and Algorithms*, 7(4), pp.329-350.

Hsu, S.A., Meindl, E.A., and Gilhousen, D.B., 1994. Determining the Power-Law Wind-Profile Exponent under Near-Neutral Stability Conditions at Sea. *Journal of Applied Meteorology*, 33(6), pp.757-772.

Khanduri, A.C., Stathopoulos, T., and Bedard, C., 1998. Wind-Induced Interference Effects on Buildings - A Review of the State-of-the-Art. *Engineering Structures*, 20(7), pp.617-630.

Kim, J., Yi, Y.K., and Malkawi, A.M., 2011. Building Form Optimization in Early Design Stage to Reduce Adverse Wind Condition - Using Computational Fluid Dynamics. In *Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association.* Sydney, pp.785-791.

Lu, S.C.-Y., Tcheng, D.K., and Yerramareddy, S., 1991. Integration of Simulation, Learning and Optimization to Support Engineering Design. *Annals of the CIRP*, 40(1), pp.143-146.

Malkawi, A.M., 2004. Developments in Environmental Performance Simulation. *Automation in Construction*, 13(4), pp.437-445.

Mueller, V. and Strobbe, T., 2013. Cloud-Based Design Analysis and Optimization Framework. In *proceedings of eCAADe 2013: Computation and Performance.* pp.185-194.

Robinson, T.D., Eldred, M., Willcox, K.E., and Haimes, R., 2008. Surrogate-Based Optimization Using Multifidelity Models with Variable Parameterization and Corrected Space Mapping. *American Institute of Aeronautics and Astronautics*, 46(11), pp.2814-2822.

Schilders, W., 2008. Introduction to Model Order Reduction. In *Model Order Reduction: Theory, Research Aspects and Applications.* Springer.

Wilkinson, S., Bradbury, G. and Hanna, S., (in press). Approximating Urban Wind Interference. In *proceedings of SimAud SpringSim'.* (Accepted for publication January 2014).

Wilkinson, S., Hanna, S., Hesselgren, L., and Mueller, V., 2013. Inductive Aerodynamics. In *proceedings of eCAADe 2013: Computation and Performance.* pp.39-48.

Veldhuizen, D.A. Van, and Lamont, G.B., 1998. Evolutionary Computation and Convergence to a Pareto Front. In *Late Breaking Papers at the Genetic Programming 1998 Conference.*

Vogl, T.P., Mangis, J.K., Rigler, A.K., Zink, W.T., and Alkon, D.L., 1988. Accelerating the Convergence of the Back-Propagation Method. *Biological Cybernetics*, 59, pp.257-263.

A.4 Wilkinson, S., Bradbury, G., and Hanna, S. (2014). Approximating Urban Wind Interference. *Simulation Series - Proceedings of SimAUD '14*, 46(7) pp.82-89.

Approximating Urban Wind Interference

Samuel Wilkinson, Gwyneth Bradbury, and Sean Hanna
University College London, UK

Abstract. A new approach is demonstrated to approximate computational fluid dynamics (CFD) in urban tall building design contexts with complex wind interference. This is achieved by training an artificial neural network (ANN) on local shape and fluid features to return surface pressure on test model meshes of complex forms. This is as opposed to the use of global model parameters and Interference Factors (IF) commonly found in previous work. The ANN is trained using shape and fluid features extracted from a set of evaluated principal (design) models (PMs). The regression function is then used to predict results based on shape features from the PM and fluid features from a one-off obstruction model (OM), context only, simulation. For the application of early-stage generative design, the errors (against CFD validation) are less than 10% centred standard deviation σ, whilst the front-end prediction times for the test cases are around 20s (up to 500 times faster than the CFD).

Keywords. Computational Fluid Dynamics (CFD), Generative Design, Machine Learning Approximation, Urban Wind Interference, Tall Buildings.

INTRODUCTION

CFD analysis in architectural design typically involves response times that are obstructive to the fast iterations of contemporary generative practice. In this parametric paradigm, architects can easily generate immense numbers of alternative scenarios but are then faced with the time-consuming task of evaluation and selection. One earlier solution focusses on early-stage design of tall buildings, using pre-computed procedural model sets, local morphological shape features, and machine learning via an artificial neural network (Wilkinson et al. 2013). It was shown that significantly faster prediction times can be achieved whilst minimising approximation errors to task-appropriate levels.

A limitation of this previous work, however, was the exclusion of surrounding context. That is, the approach treated the buildings in isolation, with unrealistically simplistic boundary conditions. In this work, the morphological features are extended with local fluid properties (upstream wind speed) to support complex urban scenarios. This is achieved by effectively superimposing an isolated building prediction of an infinitely variable generative model onto the surrounding conditions (a one-off, context-only simulation.

Many attempts have been made to approximate or generalise this kind of complex wind interference, i.e. the effect of multiple buildings in the domain (see Table 1). However, all have relied on a top-down problem definition in relating the position of identical surrounding building cuboids with a global Interference Factor (IF) for the design building. The new approach seeks to improve this significantly by: (i) allowing surrounding context of any degree of complexity; and (ii) giving vertex-level resolution rather than global effect factors. A background review, general methodology, and experimental results for two test case complexities are presented in this paper, alongside a discussion on speed and accuracy.

LITERATURE REVIEW

The background review will explore the current state of performative generative design, various methods for approximative CFD, and the state of urban wind interference with machine learning. The intention is to highlight a gap in the research, specifically on approaches to CFD approximation and interference generalisation that do not rely on global parameters or over-simplification.

Performative Generative Design

In current generative design practice, enabled by the ubiquity of computation and advances in computer aided design, integrating performance behaviours into generative models has entered the foreground (Malkawi 2004). Examples can be seen in the use of structural, energy and thermal, materiality, fabrication, and air movement (either internally for comfort and indoor air quality; or externally for structural or faade aerodynamics, pedestrian comfort, or pollution dispersal).

Air movement, predicted through CFD, suffers the most from restrictive response times, predominantly because of the historical focus on accuracy rather than speed (due to low-tolerance high-risk scenarios in aircraft and spacecraft engineering). Arguably, the margins for acceptable error are more tolerant in building design, meaning that the simulation accuracy requirements can be relaxed or traded off for speed improvements (particularly at early design stages).

In these early stages of light-weight (fast and less-accurate) performance feedback, there can be more allowance for design exploration and optimisation. This is supported by the idea of speed-accuracy trade-offs (SATs, Chittka et al. 2009), which suggests that for low-risk problems, it is often better to make faster, less accurate decisions. In other words, in the scope of the larger problem of building design, it is better to have a broader perspective on the performance variability rather than an extremely accurate but narrow perspective on fewer cases.

Approximating Computational Fluid Dynamics

CFD is one of the most intensive and time-consuming simulations in the performance assessment of tall building design. Specifically for wind analysis, it is of great importance for the safety, comfort, and efficiency of tall buildings and urban environments. Difficulty therefore typically arises in guiding massing and form decisions at early project stages due to the slow feedback from conventional CFD approaches, whilst this slow-and-accurate CFD simulation is better invested at later stages. It is therefore prudent to consider compromises in the speed-accuracy trade-off, sacrificing accuracy for speed, during these early stages so that many more design options can be explored. The need for application-specific simulation accuracy and speed that meets the demands of early design stages is proposed by Lu et al. (1991). They generate a range of reduced-order models of a combustion engine simulation, with varying accuracies and speed that can be used throughout the design process. The solution is posed as a Pareto front of non-dominated solutions, rather than a simpler trade-off curve based on biological decision making (Chittka et al., 2009).

Most approaches towards CFD approximation focus on simplification of the solver itself. For instance: simplified meshing routines; the use of lower-order discretisation; particle-based solvers; or the avoidance of turbulence models. These methods can be classed as type-one, solver approximation.

A typical example of this is the use of the 'Stable Fluids' fast fluid dynamics (FFD) solver developed by Stam (1999) for the computer graphics and games industry, which subsequently underwent some development for use in architectural practice (Chronis et al. 2011, 2012). Development and application for architectural design was motivated by three factors: a limited, low Reynolds number validation which suggested it as suitable for purposes beyond the scope of the validation (Zuo and Chen 2009, 2010); the qualitative appearance of accuracy for turbulent flows; and its remarkable speed. Zuo and Chen (2009) implemented the FFD with a zero-equation turbulence model but found that it performed worse since it was not designed or suited to the FFD approach. It should be noted, however, that with a lack of turbulence model, the solver relies on continuous interaction (such as game character movement) to compensate for numerical dissipation. The benefit is the availability of full fluid field data, although production of surface data is more difficult.

One other possible approach to this problem, type-two, is solution approximation. CFD originated in aeronautics and astronautics, as such there is a large quantity of work directed towards modelling and optimisation of airfoils, fuselages, and turbine blades. An optimisation routine will often generate large data sets of simulation data, from which knowledge of the problem can be extracted.

In one case, a large model set of turbine blades is used with a decision tree to analyse the relationship between point deformation of models and their change in surface pressure (Graening et al. 2008). Areas of high sensitivity can then be mapped onto a base geometry (preselected) and used to focus subsequent analysis. Ramanathan and Graening (2009) extend this work further to incorporate an evolutionary optimisation process, so as to use the information extracted from previous cases to create non-random initial populations of solutions and to guide the evolution.

Another example of the solution approximation approach, this time applied to building design, is by Wilkinson et al. (2013). Predictions are made through training an ANN on shape features extracted from a set of evaluated procedural tall building models.

Interference

Interference refers to the increased or decreased effect that nearby buildings may have upon the wind behaviour of one another. Within an urban situation this is very common, and since the effects can be significant it is necessary to consider the context within the simulation. That is, independently designed buildings can not be treated in isolation. Along with the large research fields of bluff bodies and computational wind engineering, interference is also a significant area of study. Research on interference is especially concerned with creating generalised recommendations, a difficult issue due to the huge variation in potential scenarios.

A common misconception is that interference always reduces wind loads from the isolated case. Whilst this may be true for a uniformed array of similar buildings in close proximity, wind loads can be increased in the more complex, realistic case. The key factors in determining the effects of interference are the size, shape, and configuration of the buildings with respect to the direction of flow. The effects have been shown to be as great as up to 46% under-prediction and 525% over-prediction from regulatory loads on simple prismatic buildings (Stathopoulos 1984). An over-prediction of wind pressure is less dangerous than an under-prediction, since the latter may cause discomfort or safety issues. Khanduri et al. (1998) present a thorough review of the full past and present state of interference. A summary of typical studies can be found in Table 1.

In all the cases shown in Table 1, simple cuboids were used with typical variables such as aspect ratio and position configuration. In other words, translating the objects over the two-dimensional horizontal plane. No studies have been performed which consider realistically complex shapes or contexts because the knowledge attained in evaluating them is typically esoteric and difficult to generalise.

A number of studies have, however, analysed the effects of a small number of adjacent structures, leading to the development of the Interference Factor (IF). This is a ratio between the wind loads with and without the interference from adjacent structures (see Table 1). In a few cases generalisation, or regression, has been attempted (the last three cases in Table 1) with the IF used as output response and basic scenario parameters as input features.

No.	Eval.	Variables	Source
2	WT	- Orient. - SD (x) - SD (y) - Aspect ratio	Agrawal et al. (2012)
2	WT + CFD	- Orientation	Zhang % Gu (2008)
2 & 3	WT	- Breadth ratio - Height ratio - Profile exponent - Configuration	Gu % Xie (2011), Xie % Gu (2004)
5	WT	- Orientation - Aspect ratio - Configuration	Jianguang (2008)
>5*	CFD	- Orientation - Configuration	Zhang et al. (2005)
No.	**Eval.**	**ANN Inputs**	**Source**
2	WT	- SD (x) - Profile exponent - Aspect ratio	English & Fricke (1999)
2	WT	- SD (x) - SD (y)	Khanduri et al. (1997)
2	WT	- Relative position - Profile exponent - Aspect ratio	Zhang & Zhang (2004)

Table 1: *Summary of selected interference sensitivity studies (No. = Number of surrounding buildings in study, WT = Wind-Tunnel, SD = Separation Distance, x = along-wind, y = across-wind). * Multiple cuboids in a small number of configurations.*

In these three cases, this method has been used along with a radial basis function (RBF) ANN. The IF was calculated from wind-tunnel data from new experiments or collected from existing literature. The limitations of this approach are the simplistic geometries (cuboids of a single height), basic configurations (typically two or three buildings), and lack of output data (only a single performance metric: the IF, rather than the varied surface pressure distribution). It should be noted that in nearly every case, the studies were constrained to a limited number of cuboids, a severe simplification in attempting to create generalised interference rules.

METHODOLOGY

The approach taken here is towards performance prediction of wind-induced surface pressure from shape analysis, developing previous work on morphological prediction (Wilkinson et al. 2013). It has previously been shown that it is possible, with a reasonable degree of accuracy and speed, to predict surface pressure for early-stage tall building design. The limitation of their work was that the models were treated in isolation without any urban context or interference: a simplification which is addressed here.

Considerable time and effort can be saved if it can be demonstrated that independent CFD simulations can be 'super-imposed' on one another with a reasonable accuracy. The end goal is to use a single, 'context-only' simulation, the Obstruction Model (OM), to make a limitless number of predictions for different designs, Principal Models (PM), using the fluid

field data alone. The clear advantage of this is that the entire OM does not need to be re-run with every change of PM.

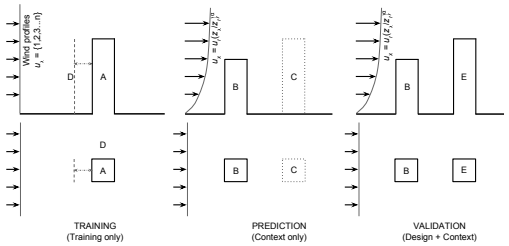


Figure 1: *Local fluid features general method. (top) elevation view, (bottom) plan - (left) Training, A=Training model (in this case = PM), D=A projected upstream; (centre) Prediction, B=OM, C=Location of PM in field; (right) Validation, B=OM, E=PM.*

The simulation of a complex urban wind environment without the PM is used as input feature for predictions on the PM in context. This is done by simulating the isolated PM under a variety of different wind speeds, and using the shape and fluid features to train an ANN and make predictions for the PM using the context-only model. The advantages of this method are in avoiding simulation of the full PM-in-context model and being able to use an existing context model to make predictions on a new PM. However, it is noted that in order to generalise this method to arbitrary PMs, a much greater training set would be required.

Simulation Methodology

The CFD solver used for the steady-state Reynolds-Averaged Navier-Stokes (RANS) simulations with a k-ϵ turbulence model is *ANSYS CFX 13.0*. Typically the models are meshed with roughly an equal number of cells (up to the maximum available computational resources), of around four million elements. The PM simulations, for the training set, therefore have a finer resolution than the OM used as the test case. The models themselves are created in *Bentley's GenerativeComponents*.

The following simulation parameters were assigned: high-resolution advection and turbulence numerics; isothermal fluid at $25°C$; a scalable wall function; a convergence residual target of $1.0e^{-6}$ RMS; and a minimum mesh edge size of $0.3m$. With these parameters, the simulations take 30±10 minutes to converge to steady-state. A transient large eddy simulation (LES) could alternatively be used instead of RANS to achieve more accurate and time-dependent surface pressures. However, due to time and resource limitations it was not possible to include a comparison in this study.

In both test cases, the wind speed is applied at an upstream inlet, with a reference speed (U_r) of $10ms^{-1}$ at a reference height (Z_r) of $10m$. The most commonly used distribution of mean wind speed with height is the 'power-law' expression:

$$U_x = U_r \cdot (Z_x/Z_r)^{\alpha} \quad (1)$$

The exponent α is an empirically derived coefficient that is dependent on the stability of the atmosphere. For neutral stability conditions it is approximately 0.143, and is appropriate for open-surroundings such as open water or landscape (Hsu et al., 1994). In the training models a constant wind profile is used, albeit with varying speeds, so as to generate a range of upstream wind speeds for every vertex.

Learning Methodology

In all cases, the learning process consists of a training and a test set of features. For a training set, $\mathbf{S}_{Tr}$, consisting of vertex feature vectors and simulated pressure extracted from the CFD, the ANN approximates the function $f^{ANN} : \mathbf{X} \rightarrow P$ where $\mathbf{X}$ is the vertex feature vector and P is the vertex pressure. $\mathbf{X}$ is defined as follows:

$$\mathbf{X} = [V_{upstream}, \mathbf{N}_{x,y,z}, \mathbf{N}\sigma^{1-5}_{x,y,z}, \mathbf{U}_{x,y,z}] \quad (2)$$

Where $V_{upstream}$ is the wind speed at the vertex's projected position upstream at a distance of $20m$ (approximately midway between PM and upstream OM), $\mathbf{N}_{x,y,z}$ are the vertex normal components, $\mathbf{N}\sigma^{1-5}_{x,y,z}$ are the vertex-ring (one through five) neighbourhood curvature (non-absolute) standard deviation components, and $\mathbf{T}_{x,y,z}$ are the normalised vertex position within the model limits. For the test prediction, $V_{upstream}$ is replaced by V, the wind speed at the vertex's position on the PM but measured in the OM fluid field.

From the 15 training set models that have been evaluated with various wind speeds, a total of 210,000 vertex features are extracted (14,000 per model), from which 10,000 are randomly selected for training the model. This number is sufficient for convergence of the ANN. An ANN with a non-linear RBF activation function is used, with a network structure of 22:20:1, i.e. 22 inputs in the feature vector, $\mathbf{X}$, 20 neurons in the hidden layer, and one output

response, Y. The error is calculated as:

$$\%Diff. = (P_{pred.} - P_{sim.})/P_{sim.range} \cdot 100 \quad (3)$$

The errors, or difference between the predicted and simulated model pressures, are reported as: the range's minimum and maximum; the mean of the absolute errors; and the standard deviation of the absolute errors (see Table 2). For both cases, a kernel density estimation is given which gives a continuous error density estimation. The smoothing kernels use a normal distribution and width of 0.1%.

RESULTS

There are two experimental cases: the first of simple geometric complexity, perhaps at the level of what may be found in the literature; and the second, of a real context and design case as might be found in practice.

Multiple Cuboid Context and Design

In the first case, five surrounding cubic buildings constitute the OM, with the PM at the centre. As training data, the PM is run independently with different wind speeds (1, 2, , 15ms-1) without any wind profile. The shape and fluid features are extracted from each of these models and used in the training set. The OM is also run, and the fluid features extracted from the appropriate positions to use as test features. Finally, the surface pressures on the PM are extracted from the full model for validation. The geometric setup of the full design and context (validation) model is shown in Figure 2. Context buildings, OM, are labelled as A and design building, PM, as B.

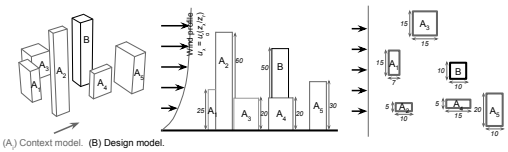


Figure 2: *Case 1 - validation model setup: (left) perspective; (centre) elevation; and (right) plan views with respective wind profiles.*

For each feature, or vertex, in the PM test case, the difference in pressure between the prediction and simulation is calculated. It is seen to converge with a mean absolute error of 6.73%, a standard deviation absolute of 4.02%, a maximum error of 33.76%, and a minimum of -26.37%. The distribution of prediction errors is shown by the kernel density estimation in Figure 3, giving a continuous error probability estimation (see Table 2 for percentiles).

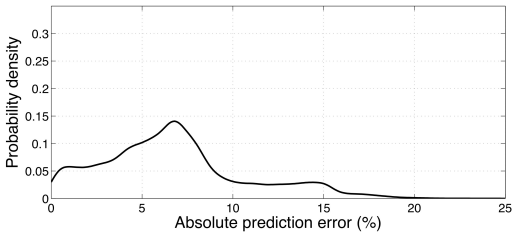


Figure 3: *Case 1 - kernel density estimation of errors.*

The difference between prediction and simulation is visualised in Figure 4: on the left is the surface pressure on the design model in the full context validation simulation; the centre is the predicted surface pressure; and on the right is the vertices % pressure difference between the two.

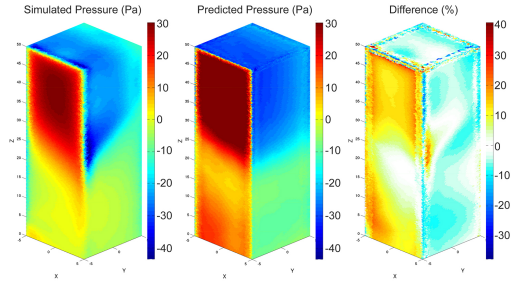


Figure 4: *Case 1 - (left) simulation; (centre) prediction; (right) error.*

Considering the complexity of the problem, this is a solid first step towards interference approximation in tall building design. The majority of the errors are less than 10 to 15% and the general pressure distribution is qualitatively correct, suggesting that the method has application in situations where accuracy can be compromised in order to facilitate rapid feedback.

Realistic Context and Design

In the second case, a realistic context model, the OM, of the dense City district in London is used (Figure 5), along with a realistic PM, design model (Figure 6), put together for the validation model (Figure 7). The PM is relatively arbitrary, but is based on prior models generated at competition, massing, or form-finding project stages. The design model is 310m tall, as compared to the upstream Swiss Re (180m) and downstream Tower 42 (183m). The wind direction is shown in Figure 8.

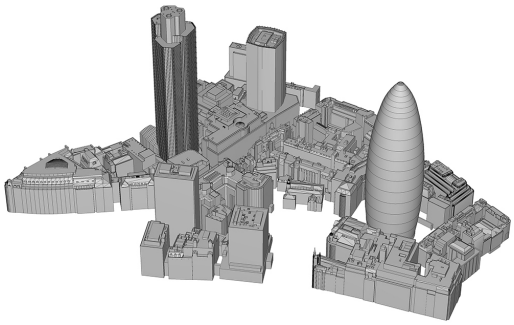


Figure 5: *Case 2 - Context only model, OM, of the City, London. Raw geometry before simplification for meshing.*

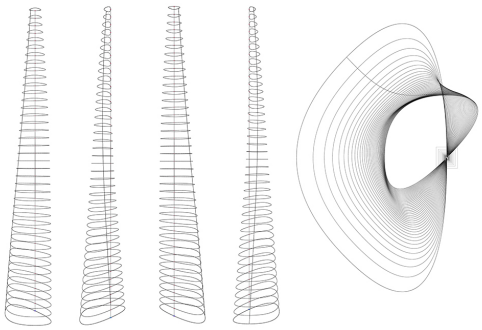


Figure 6: *Case 2 - Design model, PM - wireframe elevations and plan. Created in GenerativeComponents.*

Figure 8 shows the test and validation CFD simulation results. Note that the lower image has the design model and the change in flow streamlines is visualised. It is also interesting to note that the addition of the new design model has effects on the entire flow field, upstream and downstream. It is evident that the use of a simple, global interference factor can not do justice to the change in wind environment brought on by a new tall building.

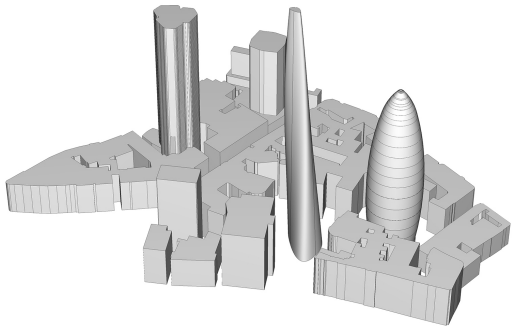


Figure 7: *Case 2 - validation model setup. After simplification for meshing. Detail resolution greater than 1m.*

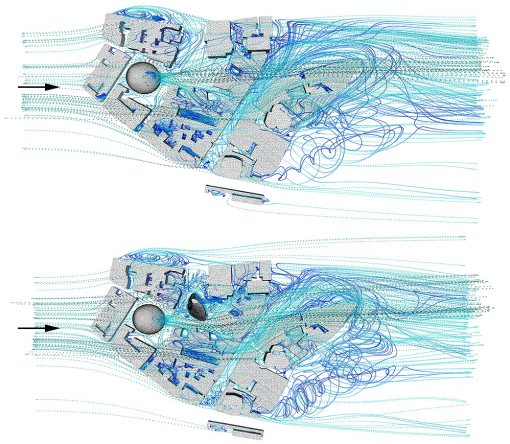


Figure 8: *Case 2 - CFD simulation, streamline visualisation: (upper) context only for test data; (lower) context and design for validation.*

The errors converge to a mean absolute of 5.80%, with a standard deviation of 9.01%, a maximum error of 11.99%, and a minimum of -55.71%. The distribution of prediction errors is shown by the kernel density estimation in Figure 9, giving a continuous distribution.

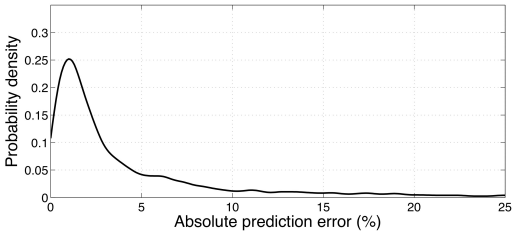


Figure 9: *Case 2 - kernel density estimation of errors.*

Figures 10 and 11 visualise the predicted surface pressures on the design model with the full context. The top image is the CFD simulation, centre the ANN prediction, and lower the % difference between simulated and predicted pressures.

DISCUSSION

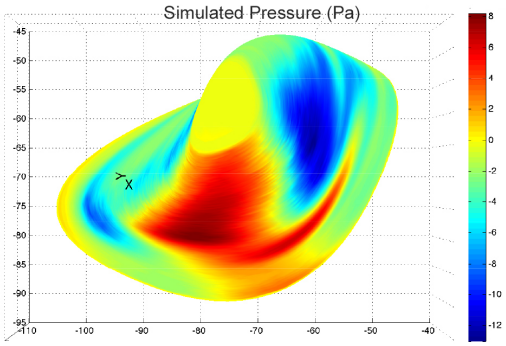
Compared to solver approximation techniques, such as the FFD solver and other low accuracy simulations, this solution approximation has the benefit of being based on a widely-used, validated CFD solver. In fact, it may be feasible to use a solver of any accuracy (such as LES or DNS, where the time improvements will be even greater). The comparative disadvantage is that the FFD can produce field rather than surface data which is useful for identifying flow patterns, assessing pedestrian comfort, and to gauge the secondary downstream effects that a new building will have on others.

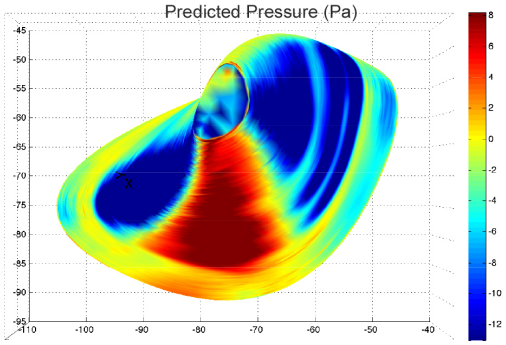
These developments represent an alternative

approach that is fundamentally different to previous attempts at interference generalisation found in the literature. The use of local features rather than global parameters allows for arbitrary complexity in the obstruction model and for vertex surface pressure visualisation rather than the global interference factor.

Response Times versus Accuracy

The errors are summarised in Table 2, and the process times for both cases and for the conventional and new approaches are given in Tables 3 and 4 respectively. Whilst the response times for the new method are approximately seven (case 1) and three (case 2) times greater than the conventional method, the true benefits are with repeatability. Therefore, in separating out the processes into front-end and back-end, the new method becomes 215 (case 1) and 510 (case 2) times faster.





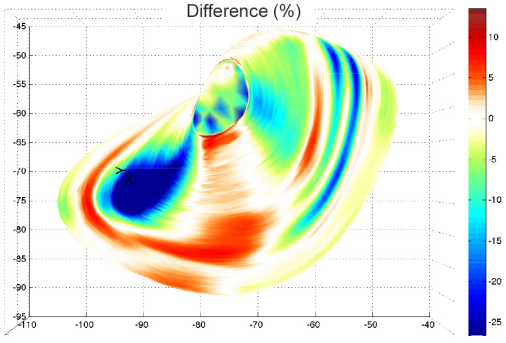


Figure 10: *Case 2 - Design model in full context, plan view. (upper) simulation; (centre) prediction; (lower) error.*

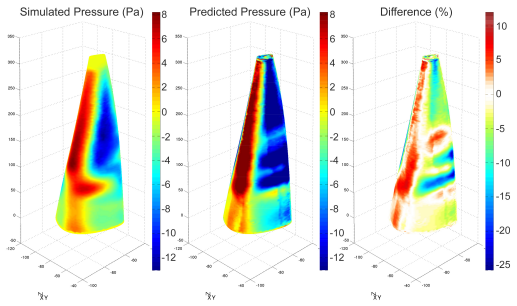


Figure 11: *Case 2 - Design model in full context. (left) simulation; (centre) prediction; (right) error.*

CONCLUSION

The methodology and results presented here demonstrate an alternative approach to urban wind interference approximation for tall building design. Through the two cases it is demonstrated that significant improvements in response time (215 and 510 times faster when comparing front-end prediction times with conventional CFD) can be made with a reasonable trade-off in accuracy (mean absolute errors of 5.8 to 6.7%) . Further improvements and generalisation can be made through the use of a procedural model to generate the training shape features, as well as through further testing on alternative models and optimisation of the training and test features.

	Case 1	Case 2
Min. / Max. range (%) *	-26.37 / 33.76	-55.71 / 11.99
Mean absolute (%)	6.73	5.80
σ absolute (%)	4.02	9.01

Table 2: *Error summary. (* worst case vertex prediction)*

Process	Case 1	Case 2
PM + OM simulation	4523	10709
Total	4523	10709

Table 3: *Conventional CFD response times.*

Off-line Process	Case 1	Case 2
PM simulations (15no.)	28535	22755
OM simulation	3793	10080
Feature extraction	300	300
ANN training	180	180
Total	32808	33315
On-line Process	**Case 1**	**Case 2**
PM feature extraction	20	20
PM prediction	1	1
Total	21	21

Table 4: *Proposed methodology response times.*

Acknowledgements

This work was sponsored by the Engineering and Physical Sciences Research Council (EPSRC), Bentley Systems Incorporated, and PLP Architecture.

References

Agrawal, N., Mittal, A.K. and Gupta, V.K. 2012. Along-Wind Interference Effects on Tall Buildings. In National Conference on Wind Engineering, India. pp. 193-204.

Chittka, L., Skorupsko, P. and Raine, N.E. 2009. Speed-Accuracy Tradeoffs in Animal Decision Making. Trends in Ecology and Evolution, 24(7), pp.400-7.

Chronis, A., Turner, A. and Tsigkari, M. 2011. Generative Fluid Dynamics: Integration of Fast Fluid Dynamics and Genetic Algorithms for wind loading optimization of a free form surface. In SimAud 2011. pp. 79-86.

Chronis, A., Tsigkari, M., David, A. and Aish, F. 2012. Design Systems, Ecology and Time. In ACADIA conference proceedings.

English, E.C. and Fricke, F.R. 1999. The Interference Index and its Prediction using a Neural Network Analysis of Wind-Tunnel Data. JWEIA, 83, pp.567-575.

Graening, L., Menzel, S., Hasenjager, M., Bihrer, T., Olhofer, M. and Sendhoff, B. 2008. Knowledge Extraction from Aerodynamic Design Data and its Application to 3D Turbine Blade Geometries. Journal of Mathematical Modelling and Algorithms, 7(4), pp.329-350.

Gu, M. and Xie, Z.-N. 2011. Interference Effects of Two and Three Super-Tall Buildings Under Wind Action. Acta Mechanica Sinica, 27(5), pp.687-696.

Hsu, S.A., Meindl, E.A. and Gilhousen, D.B. 1994. Determining the Power-Law Wind-Profile Exponent under Near-Neutral Stability Conditions at Sea. Journal of Applied Meteorology, 33(6), pp.757-772.

Jianguang, Z. 2008. Interference Effects on Wind Loading of a Group of Tall Buildings in Close Proximity. The University of Hong Kong.

Khanduri, A.C., Bedard, C. and Stathopoulos, T. 1997. Modelling Wind-Induced Interference Effects using Back-Propagation Neural Networks. JWEIA, 72, pp.71-79.

Khanduri, A.C., Stathopoulos, T. and Bedard, C. 1998. Wind-Induced Interference Effects on Buildings - A Review of the State-of-the-Art. Engineering Structures, 20(7), pp.617-630.

Lu, S.C.-Y., Tcheng, D.K. and Yerramareddy, S. 1991. Integration of Simulation, Learning and Optimization to Support Engineering Design. Annals of the CIRP, 40(1), pp.143-146.

Malkawi, A.M. 2004. Developments in Environmental Performance Simulation. Automation in Construction, 13(4), pp.437-445.

Ramanathan, S. and Graening, L. 2009. Knowledge Incorporation into Evolutionary Algorithms to Speed up Aerodynamic Design Optimizations. Universitat Stuttgart.

Stam, J. 1999. Stable Fluids. Proceedings of the 26th annual conference on Computer graphics and interactive techniques.

Stathopoulos, T. 1984. Adverse Wind Loads on Low Buildings Due to Buffeting. Journal of Structural Engineering, 110(10), 2374-2392.

Wilkinson, S., Hanna, S., Hesselgren, L. and Mueller, V. 2013. Inductive Aerodynamics. In: Stouffs, R. and Sariyildiz, S., (eds.) Proceedings of eCAADe 2013: Computation and Performance.

Xie, Z.-N. and Gu, M. 2004. Mean Interference Effects among Tall Buildings. Engineering Structures, 26(9), pp.1173-1183.

Zhang, A., Gao, C. and Zhang, L. 2005. Numerical Simulation of the Wind Field around Different Building Arrangements. JWEIA, 93(12), pp.891-904.

Zhang, A. and Gu, M. 2008. Wind Tunnel Tests and Numerical Simulations of Wind Pressures on Buildings in Staggered Arrangement. JWEIA, 96(10-11), pp.2067-2079.

Zhang, A. and Zhang, L. 2004. RBF Neural Networks for the Prediction of Building Interference Effects. Computers & Structures, 82(27), pp.2333-2339.

Zuo, W. and Chen, Q. 2009. Real-Time or Faster-than-Real-Time Simulation of Airflow in Buildings. Indoor Air, 19(1), pp.33-44.

Zuo, W. and Chen, Q. 2010. Fast and Informative Flow Simulations in a Building by using Fast Fluid Dynamics Model on Graphics Processing Unit. Building and Environment, 45(3), pp.747-757.

A.5 Wilkinson, S., Hanna, S., Hesselgren, L., and Mueller, V. (2013). Inductive Aerodynamics. In *Computation and Performance - Proceedings of the 31st eCAADe Conference*, vol.2, pp.39-48.

Inductive Aerodynamics

Samuel Wilkinson[1], Sean Hanna[1], Lars Hesselgren[2], Volker Mueller[3]
[1] *University College London, UK;* [2] *PLP Architecture, UK;* [3] *Bentley Systems, US.*

Abstract. A novel approach is presented to predict wind pressure on tall buildings for early-stage generative design exploration and optimisation. The method provides instantaneous surface pressure data, reducing performance feedback time whilst maintaining accuracy. This is achieved through the use of a machine learning algorithm trained on procedurally generated towers and steady-state CFD simulation to evaluate the training set of models. Local shape features are then calculated for every vertex in each model, and a regression function is generated as a mapping between this shape description and wind pressure. We present a background literature review, general approach, and results for a number of cases of increasing complexity.

Keywords. Machine learning; CFD; tall buildings; wind loads; procedural modelling.

Introduction

It is generally recognised that architects currently require performance information to guide their decisions almost from the inception of a project. In fact, there is a mentality present of simply trying to collect as much data as possible with the intention of synthesising it into a situated design response. This presents a problem, especially for computational fluid dynamic (CFD) wind simulation, whereby the time required to assess the performance is obstructive to the fast and iterative nature of current parametric design softwares. This is possibly due to the tendency for architectural software tools to originate in engineering fields, without due consideration of speed-accuracy tradeoffs to adjust for the application requirements (Chittka et al., 2009; Lu et al., 1991). In other words, they are typically too accurate and slow for the fast pace of modern conceptual design, massing or form decisions. Developing a method that can give real-time performance feedback about a form allows for intuitive play of the kind we are used to with physical models.

Wind engineering has traditionally been within the remit of engineers or specialists, with numerical simulation (CFD) considered a supportive tool to physical boundary layer wind tunnel (BLWT) testing. For instance, in the computational wind engineering (CWE) literature there is substantial caution around numerical analysis, namely for Reynolds-averaged Navier-Stokes (RANS) and to a lesser extent large-eddy simulations (LES) (Stathopoulos, 1997; Bitsuamlak, 2006; Dagnew et al., 2009; Menicovich et al., 2002). However, architects are increasingly getting involved with analysis, where concerns over accuracy are less paramount since demand is typically for relative scenario comparison or general flow behaviour (Lomax et al., 2001; Malkawi et al., 2005; Chronis et al., 2012).

The tall building typology has been identified as a focal area here for a number of reasons. Firstly, as height increases so too do the wind forces (along with seismic and gravitational) which has consequences on facade panelisation and structural efficiency, amongst others. We can construct a simple motivational argument to say that increased external wind force requires more opposing force, i.e. more structure, more materials, larger cores, less let-able floor space, less revenue etc. Therefore there is a need to consider the aerodynamic form of these buildings as they increase in height. Secondly, the trend for tall buildings is to build them as high as (contextually, economically and structurally) possible, necessitating cutting-edge design and construction technologies (CTBUH, 2012). Thirdly, tall building form lends itself well to parametric design as there is often a high degree of vertical logic that can be expressed neatly with mathematical expressions (this generalisation is at least more true than for shorter buildings). Given this, it is possible to easily generate a procedural, or generic, tall building model that, with a relatively small number of parameters, can represent a large number of potential designs. This becomes useful when the objective is to sample the typological space of potential buildings, which will be discussed in the methodology.

We present a novel approach to predict wind pressure on tall building models for early-stage generative design exploration and optimisation (exploration as the non-discrete parametric equivalent of

tinkering, and optimisation as the single- or multi-objective directed design space search requiring iterative testing and evaluation). The method provides fast surface pressure data with the conventional visualisation, reducing performance feedback time whilst maintaining verisimilitude.

This is achieved through the use of a machine learning algorithm, trained on a pre-computed set of CFD simulation data. ANSYS CFX 13.0, a commonly used solver in engineering practice, was used for steady-state RANS with a k-E turbulence model. The learning technique is grouped with artificial neural networks (ANN), support vector machines (SVM), and random forest (RF) decision trees, in that there is a training set of cases from which generalised rules are generated (Duffy, 1997). The term machine learning stems from the fields of computer science (Mitchell, 1997) and artificial intelligence (Samuel, 1959), but in statistics is referred to as regression and in engineering as function approximation or surrogate modelling. Once trained, this enables us to provide a new test case and make a prediction of the outcome. Inductive reasoning, epistemologically, means constructing generalisations from specific information, as opposed to deductive reasoning where small details are construed from generalisations. The fundamental outcome of this learning approach is therefore a continuous output response allowing interpolation and extrapolation between cases that have not been explicitly simulated. In doing so, we are essentially moving the simulation time from the front-end to the back-end of the process where more time is available for pre-computation.

The following section provides a review of relevant literature in the generative, performative design of tall buildings, wind modelling methods, speed-accuracy tradeoffs, incorporation of learning in design, concluding with a problem-solution hypothetical argument positioned in this state of current literature. The subsequent structure of this paper will describe the methodological approach in general terms, and results are presented from a series of experimental case studies of increasing complexity from trivial to practical. The conclusions, further work and the paper as a whole are positioned within the scope of ongoing research.

Literature Review

Tall Buildings

Tamura et al. (2009, 2010) and Tanaka et al. (2012) acknowledge the increase in tall building complexity beyond the traditional extruded rectilinear form. We are now seeing more unconventional free-style forms derived from the architect's use of more advanced modelling software. These new complicated sectional shapes that may vary with height, can actually provide better aerodynamic performance by disrupting, or 'confusing', vortex shedding and thus reducing crosswind response. Benefits can also be found in more subtle manipulations such as corner chamfering or cutting, and by creating voids, or porous regions, near the edges.

Despite rapid advances over the past century, this emerging generation of skyscrapers poses new challenges for wind engineering. Irwin (2009) discusses a number of these, such as the impact that aerodynamics have on construction cost. Since the structure itself is a large proportion of the cost, and as for tall buildings the wind is the governing lateral load, there are significant benefits to be had from reducing wind loads. This also has the effect of reducing lateral motions that can potentially cause occupant discomfort. He also suggests that shape aerodynamics must be proactively considered, and iteratively optimised, early on in the design. With the new generation of super-tall towers over 600m it is simply not possible to ignore the wind performance. He quotes a designer of the Burj Khalif, saying we practically designed the tower in the wind tunnel, and were therefore able to produce an extremely efficient aerodynamic shape that enabled the height with reasonable structural systems and costs, and without any damping system.

The increase in the use of parametric CAD softwares has seen a rise in the last decade namely with the release of Bentley GenerativeComponents and Rhino Grasshopper, plus more generally with the increased adoption of scripting. These allow the user to create parametrically associative relationships related to geometry. The extension of this idea is to use rules to define the parameters, or where these rules can be related to the performance of a model component the geometry is directed by some evaluative metric. Certain metrics can be calculated quickly without problem, but if the calculation takes time it becomes obstructive to the modelling process. We adopt the premise that it is better to have a broader range of lower resolution data rather than a limited

amount of exact data.

Speed-Accuracy Tradeoffs

Speed-accuracy tradeoffs (SATs) show that response accuracy generally increases with response time, i.e. taking more time to make a decision results in a better decision. Biological examples have been noted by Chittka et al. (2009), who explains that *'when it takes a long time to solve a difficult task, and the potential costs of errors are low, the best solution from the perspective of an animal might be to guess the solution quickly, a strategy that is likely to result in low decision accuracy.'* The two extremes can be called impulsive and reflective. This provides a neat analogy for performance analysis in design where it is necessary to consider what the application of the simulation tool is, and the consequent risks, before deciding a suitable accuracy.

Crucially though, and in conjunction with this reasoning, Burns (2005) demonstrates that making more decisions with more mistakes (fast and inaccurate) results in better overall performance (with bees, more nectar collected) than the more fastidious (slow and accurate). Defining accuracy as the proportion of choices that are correct, this highlights that accuracy should not be confined to the immediate task, i.e. simulation accuracy, but to the larger one of improving building performance (see Figure 1).

Response time is critical for performance-driven design and SATs must be considered when developing early stage tools for when large-scale decisions are made. Performance information is often scarce at this stage and iterative decisions must be made quickly, necessitating fast response times in sync with the project cycles. The development of CFD models have been focused over the past decades on improving accuracy, and computational time is optimised by specific software vendors after-the-fact, with little thought given to the accuracy required by the user. In contrast, recent developments in computer graphics have started with the desired accuracy (believable) and speed (real-time) in mind, with successful results.

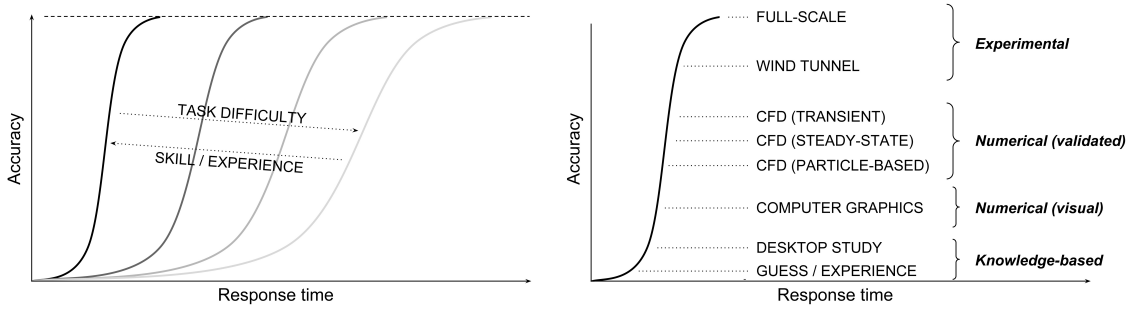


Figure 1: *(left) SAT for various task difficulties and skills; (Right) Notional positions of different modelling methods on SAT.*

In the design context, CFD can typically be used for a number of purposes: analysis of internal air movement, pollution dispersion, noise propagation, pedestrian comfort in urban environments or tall building aerodynamics. As mentioned previously, it is the last that is the focal application here, especially for early design stages. There is a paradox here, in that the most complex flow types (bluff bodies) and therefore most computationally intensive, need to be modelled in a scenario where fast results are required. The numerical method must be as accurate and fast as possible. In fact, the conclusion is reached that the fastest method has poor accuracy and the slowest the best accuracy (as would be expected, considering the speed-accuracy tradeoffs mentioned earlier). There is general agreement between (Lomax et al., 2001) and (Chronis et al., 2012) that the *'level of accuracy of a CFD simulation needs to be compromised with the turnaround time requirements of its application.'*

Lu et al. (1991) describe the same issue in mechanical engineering where slow but accurate simulation makes interactive decision making impossible, when only quick estimates are desired at early stages. It is only towards the final stages of design, *'when the engineer has converged to a small region of decision space, more accurate simulations are needed to make fine distinctions.'* The problem has therefore been present since the early 90s, but as a solution they propose integration

of simulation, optimisation and machine learning.

Inductive Learning in Design

Our approach is supported by Samarasinghe (2007), who identifies the best solution to predicting system behaviour through observational data. This is necessary when there is little or no understanding of the *'underlying mechanisms because of complex and non-linear interactions among various aspects of the problem.'* Extracting these complex relationships is often difficult since the systems are typically natural, and therefore can have randomness, heterogeneity, multiple causes and effects, and noise. Even when they are successfully extracted, they may be beyond our understanding and are held as intractable computational functions or data structures. Hanna (2011) tests the hypothesis that it is unnecessary to have any understanding of this underlying system behaviour, but rather it is possible to make predictions about the system simply by making observations. This is demonstrated by learning the structural behaviour of system components and applying them to larger-scale scenarios.

Graening et al. (2008) propose a method that allows the extraction of comprehensible knowledge from aerodynamic design data (jet-blades) represented by discrete unstructured surface meshes. They use a displacement measure in order to investigate local differences between designs and the resulting performance variation. Knowledge, or rule, extraction from CFD data is primarily used to guide human- centred design by improving understanding of the system's behaviour, whether it is for jet turbine blade optimisation or architectural design. Whilst the connection between local geometric features and surface pressure has been extended and changed here, and used for a different application, this work is a close precedent.

Problem Hypothesis

It is argued here that approximations of CFD simulations can be made with machine learning regression, using geometric shape descriptors as the learning features. The entire evaluation process can be broadly split into five key work areas: i) procedural geometry generation; ii) batch simulation; iii) shape feature generation; iv) machine learning training; v) prediction and visualisation. Feature generation is essentially the core of the process since the solution depends heavily on geometric description so as to define surface pressure as a function of it. We hypothesise that surface pressure distribution arising from wind flow around tall buildings can be learnt and predicted with an accuracy appropriate to early stage design (feedback from practice indicates <20% error) using shape feature description. It can be shown that it is possible to combine, with an acceptable error, methods that have the separate contradictory objectives of predictive accuracy and speed.

Methodology

Data Set Generation: Procedural Modelling

The parametric model was created in Bentley GenerativeComponents. The goal was to create a generalised tower model, with the two properties of minimising the number of parameters used whilst maximising the design representation potential, i.e. the number of possible buildings it could create. This is important when considering optimisation or exploratory design space searches to avoid the curse of dimensionality. This means that as the number of variables increases, the design space increases exponentially by nD, where n is the number of samples taken per parameter and D is the number of parameters, or dimensionality. There is therefore clearly a compromise to be made between model efficiency and represent-ability.

The geometry for the training set was generated using a procedural tall building model with a select number of key parameters. There are in fact three separate topologies in the procedural model with their own parameters, since it is difficult to incorporate the entire design space with one parametric logic (Park et al., 2004; Samareh, 1999). Using the unstructured triangulated surface mesh from these means we are not limited by a single parametric topology in the learning phase of the method (Graening et al., 2008). Local surface-mesh shape characteristics are used as input features to the learning algorithm instead of the design parameters, avoiding reliance on any

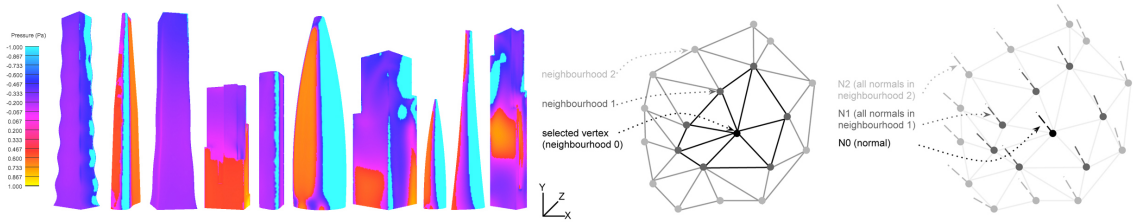


Figure 2: *(left) Examples of evaluated procedural models in the training set on Case 4; (Right) Mesh feature extraction.*

one parametric model definition.

Simulation Method

An established solver, ANSYS CFX 13.0, was used throughout to run the RANS steady-state simulations, with a k-ϵ turbulence model as it is regarded as the most robust. Each simulation, depending on the complexity, requires up to 60 minutes to converge (on a 2.66GHz i7). Solver convergence is reached when residuals fall below a minimum of $1e^{-6}$, typically at around 100 to 200 iterations. The number of cells in the tetrahedral meshes varies between 0.8x10^6 and 1.5x10^6 depending on the geometry, with prismatic expansion on surfaces 3 cells deep and a minimum cell size of $0.1m$. The wind was applied at an upstream inlet, with a reference speed (U_r) of $1ms^{-1}$ at a reference height (Z_r) of $10m$. The most commonly used distribution of mean wind speed with height is the 'power-law' expression:

$$U_x = U_r \cdot (Z_x/Z_r)^{\alpha} \tag{1}$$

The exponent α is an empirically derived coefficient that is dependent on the stability of the atmosphere. For neutral stability conditions it is approximately 0.143, and is appropriate for open-surroundings such as open water or landscape. Future work will include a wind profile that takes surrounding surface roughness, or context, into account, as well as potential wind direction change with height.

$$P(Z, N_{(x,y,z)}, N\sigma^{1-5}_{(x,y,z)}, U_{(x,y,z)}) \tag{2}$$

For a specific model vertex, P is the surface pressure, Z is the height, $N_{(x,y,z)}$ are the normal components, $N\sigma^{1-5}_{(x,y,z)}$ is the standard deviation σ of normal components of cumulative mesh neighbourhood rings 1 through 5, and $U_{(x,y,z)}$ are the normalised model position components. The extent of the neighbourhood curvature can be extended beyond 5 rings, within computational resource limits. The definition in Equation 2 gives 22 inputs and 1 output feature to train the learning algorithm for all cases described below.

For the Orientation, Height and Topology cases, an Artificial Neural Network (ANN) was used, with a 70:30% split of the provided data to training:validation. For the first two cases, separate sets constituting entire models were also held back for testing, i.e. training was at 15° and $20m$ intervals respectively. For the third case, there was no extra test set but the whole was split 70:15:15% to training:validation:test. Validation data is to check for convergence during training. For the fourth case, training data was from the procedural tall building model and test data from another set of real buildings. In this case, a Random Forest (RF) algorithm was used instead as it provided better results for the more complex problem. Further work is needed with both methods to understand their applicability to certain tasks, however it is known that the RF is better with noisy data sets than the ANN. Training set sizes and summary results are given in Table 1, and computation times are given in Table 2.

Shape Features and Learning

This method creates a definition for the pressure at a point on the model as the function of a local geometric description. To describe a simple example of the process: there are N models of a cuboid with various orientations; each is evaluated, and the pressure P is extracted at M

points over each model; for every M, a shape descriptor X is calculated, such as the vertex height, normal components, curvature, etc; this gives a set of geometric characteristics, and a corresponding pressure value; these sets of $P(X)$ are used as the training data. Pressure distribution is predicted from these geometric descriptors alone meaning the selection is critical. A sensitivity analysis has been conducted with a variety of descriptors to determine suitable representation, details of which are not included here. When a new case is presented, the shape descriptors are calculated and used to make a prediction of P. The feature definition for point pressure in $\mathbf{R}22$ vector space used throughout the following is:

Results

Cuboid Orientation

The first and most simple test is the rotation of a cuboid, of width and depth 10m, and height 50m. Simulations were run at 5° intervals from 0 to 85°, and the ANN trained on 15° and tested at 5° intervals. The sensitivity analysis here varies the number of training samples and measures the standard deviation, σ, of the difference between simulation and prediction. Figure 3 (left) shows the error σ against orientation for various set sizes (bold vertical lines are training intervals of 15°), (centre) the training regression of the entire set, and (right) the prediction error for an orientation of 25°. With less training data, it can be seen that error is highest around 45° when flow bifurcations (regime change) occur, although this is negated with sufficient data.

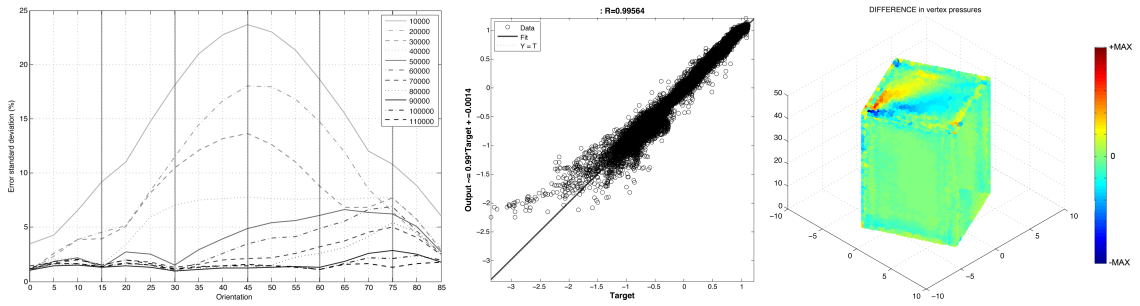


Figure 3: *(left) Orientation vs. Error σ %; (Centre) Training set regression, R=0.99564; (Right) Prediction error (25°).*

Cuboid Height

Secondly, a parametric cuboid was created with width and depth $10m$, and height varying from 10 to $100m$ in $5m$ increments. Figure 4 (left) shows the variability when trained on 10, 20, 30 and $45m$ intervals, and (right) the prediction error for a height of $25m$ when trained at $20m$ intervals.

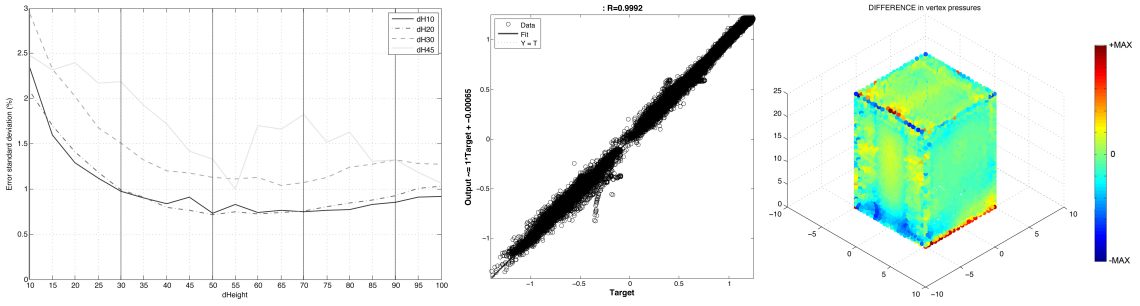


Figure 4: *(left) dHeight vs. Error σ %; (Centre) Training set regression, R=0.9992; (Right) Prediction error (25m).*

Topology

Here the number of edges was varied from 3 to 10, with 0 (circle), diameter $10m$ and height $50m$. Instead of keeping a complete model separate for testing as in the last two cases, here all cases were used but only a fraction of the total data set was used. This is varied in Figure 5 (left), with a training set ranging from 10000 to 50000.

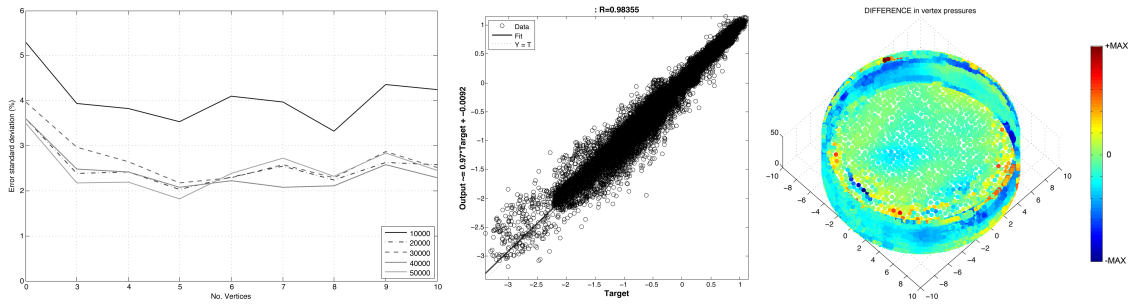


Figure 5: *(left) No. Edges vs. Error σ %; (Centre) Training set regression, R=0.98355; (Right) Prediction error (n0).*

Tall Buildings

In the final case, training data was collected from simulations of 600 procedural tall building models, with a total of over 4x10^6 shape features extracted. This was down-sampled to 10^5 by removing features in close proximity to reduce training time. The test set contains 10 real tall buildings from around the world, selected for their range of unique architectural characteristics. Figure 6 below shows predicted surface pressure distribution in the top row, and the error distribution for the set in the bottom row. The pressure range (-5.5 to 2.0 Pa) was taken over the entire test set, as was the absolute error range (0 to 65.2%). The error distribution is shown in Figure 7 (right), which fits a Gaussian normal distribution. Error percentiles: 99th = 35.7%, 95th = 20.0%, 90th = 13.0%, 75th = 6.1%. That is, 75% of the test features have an error below 6.1%.

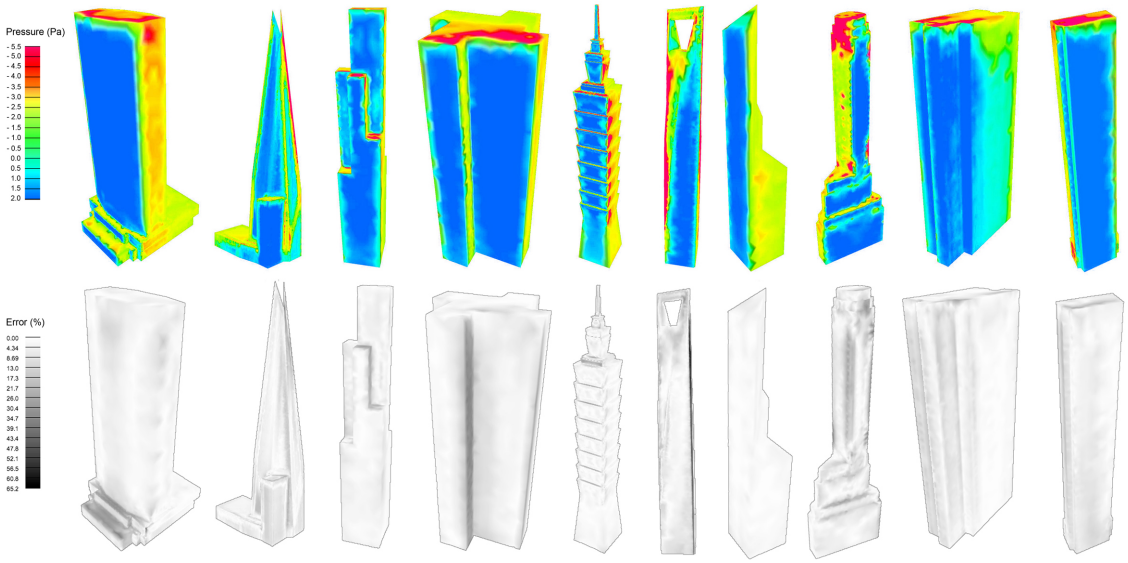


Figure 6: *(upper) Predicted pressure, Pa; (lower) Error, %. Pressure range is the min. and max. of the entire set for comparison, the error range is absolute max. error of the set (65.2%).*
(left to right) (1) Metlife Building, NYC; (2) The Shard, London; (3) Willis Tower (Sears), Chicago; (4) Euston Tower, London; (5) Taipei 101, Taiwan; (6) Shanghai World Financial Centre; (7) Bank of China; (8) Exchange Place, NYC; (9) Frankfurter Buro Centre, Frankfurt; (10) Washington Street, NYC.

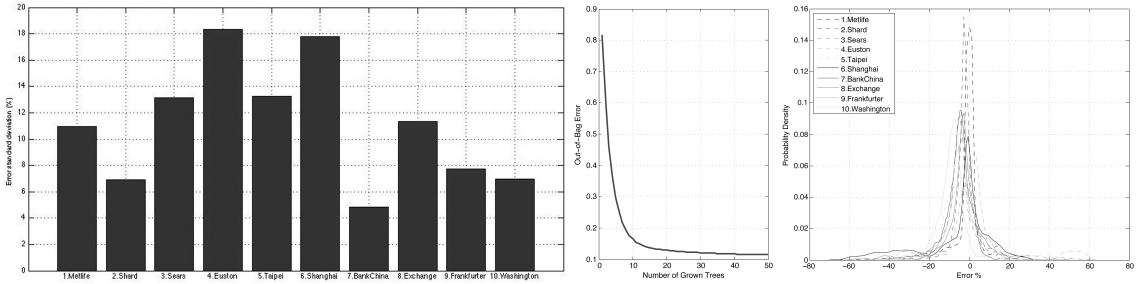


Figure 7: *(left) Error σ % for each case; (Centre) Random Forest learning convergence; (Right) Error probability density.*

Results Summary

Case	Min σ error (%)	Max σ error (%)	Training set size
Orientation	1.2 (55°)	1.6 (10°)	110000 (15° training intervals)
Height	0.7 (10m)	2.0 (50m)	44720 (20m training intervals)
Topology	1.8 (5 Edges)	3.5 (0 Edges)	50000
Real	4.8 (Bank of China)	18.3 (Euston)	100000 (Procedural training)

Table 1: *Summary of minimum and maximum error standard deviations (% over test pressure range).*

Case	Train Sim.	Train Feat. Gen.[b]	Train	Predict Feat. Gen.[a]	Predict
Orientation	21600	9060	2600	1540	< 0.1
Height	18000	2370	720	620	< 0.1
Topology	32400	4670	1060	1750	< 0.1
Real	2160000	12000	620	720	< 0.1

Table 2: *Summary of time [seconds] required for each case, split into Training (one-off back-end time) and Prediction (front-end time). Mean feature generation time is 0.085s/vertex. [a]Mean over all test set. [b]After down-sampling.*

Conclusion

The results show that it is possible to achieve a relatively small prediction error (Figure 7 and Table 1) for less time (Table 2), with the methodology and constraints described. These prediction errors are necessary for the compromise in avoiding considerably intensive CFD simulation. Traditionally, for every individual CFD simulation the process can take a minimum of 1 hour, compared to our methodology that has a total front-end prediction time of under 12 minutes (for feature generation and prediction) and a back-end, one-off training set simulation time of 600 hours (for the real case). Once trained, an unlimited number of predictions can then be made.

Whilst these preliminary results are outside the rigorous accuracy necessary for final engineering analysis, they are within the boundaries acceptable for early-stage concept design for tall buildings, where interactive response time is a significant consideration. The prediction accuracy and response times achieved are promising for further work given the well-known complexities of fluid behaviour.

The next stages of the work are to consider time-dependent simulations to fully consider the approximation of turbulence, vortex shedding and gusts, as well as interference from complex urban contexts on boundary conditions, and further improvement to the shape feature selection and generation time.

Acknowledgements

This research was sponsored by the EPSRC, Bentley Systems and PLP Architects.

References

Bitsuamlak, G., 2006. *Application of Computational Wind Engineering: A Practical Perspective.* In Third National Conference in Wind Engineering. pp. 1-19.

Burns, J.G., 2005. *Impulsive Bees Forage Better: The Advantage of Quick, Sometimes Inaccurate Foraging Decisions.* Animal Behaviour, 70(6), pp.1-5.

Chittka, L., Skorupski, P. & Raine, N.E., 2009. *Speed-Accuracy Tradeoffs in Animal Decision Making.* Trends in ecology & evolution, 24(7), pp.400-7.

Chronis, A. et al., 2012. *Design Systems, Ecology and Time.* In ACADIA.

CTBUH, 2012. *Tall Buildings in Numbers : A Tall Building Review*, 2012(1).

Dagnew, A.K., Bitsuamlak, G. & Merrick, R., 2009. *Computational Evaluation of Wind Pressures on Tall Buildings.* In 11th Americas Conference on Wind Engineering.

Duffy, A., 1997. *The What and How of Learning in Design.* IEEE Expert, 12(3), pp.71-76.

Graening, L. et al., 2008. *Knowledge Extraction from Aerodynamic Design Data and its Application to 3D Turbine Blade Geometries.* JMMA, 7(4), pp.329-350.

Hanna, S., 2011. *Addressing Complex Design Problems through Inductive Learning.*

Irwin, P.A., 2009. *Wind Engineering Challenges of the New Generation of Super-Tall Buildings.* JWEIA, 97(7-8), pp.328-334.

Lomax, H., Pulliam, T.H. & Zingg, D.W., 2001. *Fundamentals of Computational Fluid Dynamics,* Berlin: Springer.

Lu, S.C.-Y., Tcheng, D.K. & Yerramareddy, S., 1991. *Integration of Simulation, Learning and Optimization to Support Engineering Design.* Annals of the CIRP, 40(1), pp.143-146.

Malkawi, A.M. et al., 2005. *Decision Support and Design Evolution: Integrating Genetic Algorithms, CFD and Visualization.* AIC, 14(1), pp.33-44.

Menicovich, D. et al., 2002. *Generation and Integration of an Aerodynamic Performance Database within the Concept Design Phase of Tall Buildings.*

Mitchell, T.M., 1997. *Machine Learning*, McGraw-Hill.

Park, S.M. et al., 2004. *Tall Building Form Generation by Parametric Design Process.* In CTBUH. Seoul Conference, pp. 1-7.

Samarasinghe, S., 2007. *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition,* Auerbach Publications, NY.

Samareh, J.A., 1999. *A Survey of Shape Parameterisation Techniques.* In CEAS/AIAA/ICASE/NASA Langley International Forum on Aeroelasticity and Structural Dynamics. pp. 333-343.

Samuel, A.L., 1959. *Some Studies in Machine Learning using the Game of Checkers.* IBM JRD, 3(3).

Stathopoulos, T., 1997. *Computational Wind Engineering: Past Achievements and Future Challenges.* JWEIA, 67-68, pp.509-532.

Tamura, Y., 2009. *Wind and Tall Buildings.* In EACWE 5.

Tamura, Y. et al., 2010. *Aerodynamic Characteristics of Tall Building Models with Various Unconventional Configurations.* Structures Congress 2010, pp.278-278.

Tanaka, H. et al., 2012. *Experimental Investigation of Aerodynamic Forces and Wind Pressures Acting on Tall Buildings with Various Unconventional Configurations.* JWEIA, 107-108, pp.179-191.

Appendix B

Code

B.1 GC-FFD Node (*C#*)

```
using System;
using OpenTK;
using graph = OpenTK.Graphics;
using OpenTK.Graphics.OpenGL;
using OpenTK.Input;

using Bentley.Geometry;
using Bentley.Geometry.Mesh;
using Bentley.GenerativeComponents.UISupport;
using Bentley.GenerativeComponents;
using Bentley.GenerativeComponents.GCScript;
using Bentley.GenerativeComponents.MicroStation;

namespace Bentley.GenerativeComponents.Features.Specific { // Must be in this namespace.
    [HideInheritedTechniques]
    public class fluidSolver : Mesh {
        [Technique]
        public bool FromMesh(
            FeatureUpdateContext updateContext,
            Mesh mesh,
            [Optional, DefaultValue(40)] int domainXDim,
            [Optional, DefaultValue(40)] int domainYDim,
            [Optional, DefaultValue(70)] int domainZDim,
            [Optional, DefaultValue(1.0)] double meshScaleFactor,
            [Optional, DefaultValue("meshData.txt")] string meshDataFileName,
            [Optional, DefaultValue(false)] bool continuousRun,
            [Out] ref double maxSurfPressure
            ) {
            System.IO.StreamWriter file =
                new System.IO.StreamWriter("c:/_FFDsolver/" + meshDataFileName);

            int vcount = mesh.Vertices.GetLength(0);
            int fcount = mesh.Indices.GetLength(0);

            string lines = "vcount " + vcount;
            file.WriteLine(lines);

            for (int i = 0; i ¡ vcount; i++){
                lines = mesh.Vertices[i].X + "," + mesh.Vertices[i].Y + "," + mesh.Vertices[i].Z;
                file.WriteLine(lines);
            }

            lines = "fcount " + fcount;
            file.WriteLine(lines);

            for (int i = 0; i ¡ fcount; i++) {
                lines = mesh.Indices[i][0] + "," + mesh.Indices[i][1] + "," + mesh.Indices[i][2];
                file.WriteLine(lines);
            }

            file.Close();

            string filename = "C:/_FFDsolver/" + meshDataFileName;
            string consolevars = filename + " " + domainXDim + " " + domainYDim + " " + domainZDim +
                " " + meshScaleFactor + " " + continuousRun;
```

```
            ProcessStartInfo si = new ProcessStartInfo("C:/_FFDsolver/120202.exe", consolevars);
            Process p = Process.Start(si);

            p.EnableRaisingEvents = true;
            p.Exited += new EventHandler(p_Exited);
            p.WaitForExit();

            double mst;
            try {
                using (StreamReader sr = new StreamReader("C:/_FFDsolver/outData.txt")) {
                    string line;
                    string[] words;
                    if ((line = sr.ReadLine()) != null) {
                        words = line.Split(',', ' ');
                        mst = Convert.ToDouble(words[0]);
                    }
                    else mst = 0.0;
                }
            }

            catch (Exception ef) {
                Console.WriteLine("FAIL");
                Console.WriteLine(ef.Message);
                mst = 0;
            }

            maxSurfPressure = mst;
            return true;
        }

        void p_Exited(object sender, EventArgs e) { }
    }
}
```

B.2 Standard Deviation Calculation (*Java*)

```
//Note: See Figure 5.37.

//stdev function called in the following calc function.
//Requires list of neighbours and the central vertex.

void stdev(ArrayList a, int ve) {
    ArrayList vN = new  ArrayList();
    ArrayList vV = new ArrayList();
    for (int i=0; i<a.size(); i++) {
        int d = (integer) a.get(i);
        vN.add((PVector) norms.get(d));
        vV.add((PVector) verts.get(d));
    }

    PVector b = new PVector(0, 0, 0);          //b is the mean of the normals in ring N.
    PVector c = new PVector(0, 0, 0);          //c is the standard deviation of the x, y, z components.
    PVector ver = (PVector) verts.get(ve);

    float ondist=0;
    float offdist=0;
    PVector zerov = (PVector) vV.get(0);
    PVector zerovN = (PVector) vV.get(0);
    PVector zeronN = (PVector) vN.get(0);
    PVector zerovNoff = new PVector(zerovN.x+zeronN.x,zerovN.y+zeronN.y,zerovN.z+zeronN.z);

    float disCo = 0;

    for (int i=0; i<a.size(); i++) {          //for every vertex in the ring
        PVector bT = (PVector) vN.get(i);
        PVector bTV = (PVector) vV.get(i);
        float dis = 1.0 / bTV.dist(ver);
        disCo += dis;
        b.x += bT.x * dis;
        b.y += bT.y * dis;
        b.z += bT.z * dis;

        if(i>0) ondist += bTV.dist(zerov);
        zerov = bTV;
```

```
        if(i>0) {
            PVector zerovNoffCurrent = new PVector(bTV.x+bT.x,bTV.y+bT.y,bTV.z+bT.z);
            offdist += zerovNoffCurrent.dist(zerovNoff);
        }
        zerovNoff = new PVector(bTV.x+bT.x,bTV.y+bT.y,bTV.z+bT.z);
    }

    b.div(disCo);

    for (int i=0; i<a.size(); i++) {
        PVector cT = (PVector) vN.get(i);
        PVector bTV = (PVector) vV.get(i);
        float dis = 1.0 / bTV.dist(ver);
        c.x += pow(((cT.x)-b.x), 2) * dis;
        c.y += pow(((cT.y)-b.y), 2) * dis;
        c.z += pow(((cT.z)-b.z), 2) * dis;
    }

    c.div(disCo);
    c.x=sqrt(c.x);          //sqrt to convert from variance to stdev.
    c.y=sqrt(c.y);
    c.z=sqrt(c.z);

    //stdevSave
    if(offdist>=ondist) stdev.add(new PVector(c.x, c.y, c.z));          // If convex, keep stdev +ve.
    else stdev.add(new PVector(-c.x, -c.y, -c.z));          // If concave, make stdev -ve.
}
```

B.3 Feature Calculation (*Java*)

```
//For every vertex, v:
void calc(int v) {
    //N0, the vertex index itself
    stdev = new ArrayList();
    n = new ArrayList[NN];          //n is the list of indices per ring.
    for (int i=0; i<NN; i++) n[i] = new ArrayList();
    n[0].add((int)v);

    //N1, first ring
    for (int i=0; i<inds.size(); i++) {
        PVector tI = (PVector) inds.get(i);          //get 3 indices of that faces.
        if (tI.x == v) {
            n[1].add((int)tI.y);
            n[1].add((int)tI.z);
        }
        if (tI.y == v) {
            n[1].add((int)tI.x);
            n[1].add((int)tI.z);
        }
        if (tI.z == v) {
            n[1].add((int)tI.x);
            n[1].add((int)tI.y);
        }
    }
    HashSet hs = new HashSet();
    hs.addAll(n[1]);
    n[1].clear();
    n[1].addAll(hs);          //duplicate indices removed.

    //Nearest neighbours
    for (int i=2; i<NN; i++) {          //for every subsequent ring
        for (int j=0; j<n[i-1].size(); j++) {          //for the number of the previous ring's indices
            int tJ = (integer) n[i-1].get(j);          //get list of previous indices
            for (int k=0; k<inds.size(); k++) {          //run through all indices
                PVector tI = (PVector) inds.get(k);          //get 3 indices of that faces
                if (tI.x==tJ || tI.y==tJ || tI.z==tJ) {          //if any of the indices on the mesh match
                    n[i].add((int)tI.x);          //add it
                    n[i].add((int)tI.y);
                    n[i].add((int)tI.z);
                }
            }
        }
```

```
        hs = new HashSet();
        hs.addAll(n[i]);
        n[i].clear();
        n[i].addAll(hs);          //duplicate indices removed.
    }

    //remove duplicates across neighbours, for independent rings.
    for (int i=0; i<NN; i++) {
        for (int j=i+1; j<NN; j++) {
            for (int k=0; k<n[i].size(); k++) {
                for (int l=0; l<n[j].size(); l++) {
                    int a = (integer) n[i].get(k);
                    int b = (integer) n[j].get(l);
                    if (a==b) n[j].remove(l);
                }
            }
        }
    }

    for (int i=1; i<NN; i++) stdev(n[i], v);

    PVector vN = (PVector) norms.get(v);
    PVector vV = (PVector) verts.get(v);
    float xpos = (vV.x - minRange.x) / (maxRange.x - minRange.x);
    float ypos = (vV.y - minRange.y) / (maxRange.y - minRange.y);
    float zpos = (vV.z - minRange.z) / (maxRange.z - minRange.z);
    float p = (float) pre.get(v);
    PVector [] nn = new PVector[NN-1];
    for (int i=0; i<NN-1; i++) nn[i] = (PVector) stdev.get(i);


    //PRINT TO FILE
    output.println(vV.z+","+vN.x+","+vN.y+","+vN.z+","+nn[0].x+","+nn[0].y+","+nn[0].z+","+
        nn[1].x+","+nn[1].y+","+nn[1].z+","+nn[2].x+","+nn[2].y+","+nn[2].z+","+
        nn[3].x+","+nn[3].y+","+nn[3].z+","+nn[4].x+","+nn[4].y+","+nn[4].z+","+
        xpos+","+ypos+","+zpos+","+p);
}
```

B.4 Machine Learning (*Matlab*)

```
//Note: Code for cuboid rotation study, see Section 5.3.8.

//TRAINING
trainData = [data00; data15; data30; data45; data60; data75];          //Load training data
n = size(trainData,1);          //Get full available training set size
n2=10000;          //Specify desigray training set size

indices = randperm(n);          //Randomise
trainDataX = trainData(indices(1:n2),1:22);          //Separate training inputs...
trainDataY = trainData(indices(1:n2),23);          //...and training output

hiddenLayerSize = 20;
net = fitnet(hiddenLayerSize);
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;
[net,tr] = train(net,transpose(trainDataX),transpose(trainDataY));          //Train


//GENERAL TESTING (over all remaining data)
testData = [data05; data10; data20; data25; data35; data40,...          //Load remaining test data
data50; data55; data65; data70; data80; data85];

testDataX = transpose(testData(indices(n2+1:n),1:22));          //Separate test inputs...
testDataY = transpose(testData(indices(n2+1:n),23));          //...and test output
pgrayictY = net(testDataX);          //Make test pgrayictions
d = pgrayictY-testDataY;          //Calculate the raw difference

data= [trainData; testData];          //All data for full pressure range

dMIN = min(d)/range(data(:,end))*100;
dMAX = max(d)/range(data(:,end))*100;
dSTD = std(abs(d))/range(data(:,end))*100;
```

```
dMEAN = mean(abs(d))/range(data(:,end))*100;

//SPECIFIC TESTING (for a selected model)
testX05 = transpose(data05(1:end,1:(end-1)));
testY05 = transpose(data05(1:end,end));
pgrayictY05 = net(testX05);
d05 = pgrayictY05-testY05;

//RESULTS VISUALISATION (for a selected model)
mesh05 = csvread('mesh05.csv',6);            //Load test model mesh
pts = mesh05(1:end,1:3);
connects = csvread('mesh05.csv', size(dpts,1)+8)+1;
TR = TriRep(connects,pts(:,1),pts(:,2),pts(:,3));            //Reconstruct mesh topology

c = pgrayictY05/range(data(:,end));            //Get pressure range, mapped to full set range
trisurf(TR,c,'EdgeColor','none');            //Colour mesh
caxis([min(pgrayictY05)/range(data(:,end)), max(pgrayictY05)/range(data(:,end)))
```

B.5 LES Pressure Range Extraction (*Java*)

```
ArrayList [] p;
float mini, maxi;
int a = 0;
int b = 10000;

void findMinMax() {
    String file = "85";
    String[] lines;
    outputMax = createWriter("max"+file+".csv");
    outputMin = createWriter("min"+file+".csv");
    p = new ArrayList[36];
    mini = 1000;
    maxi = -1000;

    for (int j=0; j<=35; j++) {
        p[j] = new ArrayList();
        lines = loadStrings(file+"_"+j+".csv");
        for (int i=6; i<lines.length; i++) {
            String[] words = split(lines[i], ",");
            p[j].add(float(words[0]));
        }
    }

    for (int i=0; i<p[0].size(); i++) {
        mini = 1000;
        maxi = -1000;
        for (int j=12; j<=35; j++) {
            float pT = (float) p[j].get(i);
            if (pT>maxi) maxi=pT;
            if (pT<mini) mini=pT;
        }
        outputMax.println(maxi);
        outputMin.println(mini);
    }

    outputMax.flush();
    outputMin.flush();
    outputMax.close();
    outputMin.close();
    exit();
}
```

B.6 Procedural Model (*GC*)

```
transaction modelChange 'Add tower genny' {
    node User.Objects.tower Bentley.GC.Features.Solid{
        Technique = 'ByFunction';
```

```
Function = function (CoordinateSystem cs){
    double faceCounter = 0;
    Solid s3;
    while(faceCounter!=19){
        double sel = Random(0,2);

        s3 = new Solid(this);
        CoordinateSystem csR = new CoordinateSystem();

        if(sel==0){
            int N;
            double W, D, H, mSF, tSF, F, o;

            while(s3.Success==false) {
                double x = {};
                double y = {};
                double z = {};
                N = Random(3,7); //Par5
                H = Random(100*10,200*10)/10; //Par3
                mSF = Random(5,11)/10; //Par7
                tSF = Random(1,11)/10; //Par8
                while(tSF>=mSF) tSF = Random(1,11)/10;
                double Hsf = {1,mSF,tSF};
                F = Random(3,51)/10; //Par6
                o = Random(2,4); //Par9

                double erX;
                double erY;
                if(N==3) {
                    erX = 1/(0.43301270189222*2);
                    erY = 1/0.750;
                }
                if(N==4){
                    erX = 1/(0.35355339059328*2);
                    erY = 1/(0.35355339059328*2);
                }
                if(N==5){
                    erX = 1/(0.951056516295154);
                    erY = 2/(0.809016994374947+1);
                }
                if(N==6){
                    erX = 1/1;
                    erY = 1/(0.43301270189222*2);
                }

                W = Random(100,600)/10; //Par1
                D = Random(100,600)/10; //Par2

                for(int i=0; i<3; i++){
                    x[i] = {};
                    y[i] = {};
                    z[i] = {};
                    for(int j=0; j<N; j++){
                        x[i][j]=Sin((360/N)*j+(180/N))*erX*0.5*W *Hsf[i] ;
                        y[i][j]=Cos((360/N)*j+(180/N))*erY*0.5*D *Hsf[i] ;
                        z[i][j]=i*(H/2);
                    }
                }


                csR.ByUniversalTransform(cs,0,0,0,0,0,Random(0,180*10)/10); //Par4
                Point p = new Point();
                p.ByCartesianCoordinates(csR,x,y,z);
                Curve c = new Curve();

                int selo = Random(0,4);
                /*fillet*/ if(selo==0) {
                    c.FromPointSet(p,CurveFromPointSetOption.LineFilletComposite,F,null,true);
                }
                /*oneEdge*/ if(selo==1) {
                    c.FromPointSet(p,CurveFromPointSetOption.BSplineCurveByPoints,null,null,true);
                }
                /*smooth*/ if(selo==2) {
                    c.FromPointSet(p,CurveFromPointSetOption.BSplineCurveByPoles,null,null,true);
                }
                /*polygon*/ if(selo==3) {
```

```
            c.FromPointSet(p,CurveFromPointSetOption.PolyLine,null,null,true);
        }
        BSplineSurface b = new BSplineSurface();
        b.LoftCurves(c,o);
        Solid s1 = new Solid(this);
        s1.BySurfaceCapping(b);
        CoordinateSystem csT = new CoordinateSystem();
        csT.ByCartesianCoordinates(cs,0,0,H);
        Solid s2 = new Solid();
        s2.SlabAtCentroid(csT,150,200,H*2);
        s3.BooleanOperation(s2,s1,BooleanOperation.Difference);
    }

    //Print(W+","+D+","+H+","+csR.ZRotation
        +","+N+","+F+","+mSF+","+tSF+","+o);
}

if(sel==1){
    double AMP = Random(1,100)/10.0;
    double DECAY = Random(100,1000);
    double FREQ = Random(1,20)/10.0;
    double FILLET = Random(5,200)/10.0;
    double FREQOFF = Random(-180,180);

    double W = Random(10,20);
    double D = Random(10,20);
    double Hei = Random(100,200);

    CoordinateSystem csRo = new CoordinateSystem();
    csRo.ByUniversalTransform(cs,0,0,0,0,0,Random(0,180*10)/10);

    Point pt1 = new Point();
    pt1.ByCartesianCoordinates(csRo,{-W,-W,W,W},{-D,D,D,-D},0);

    Curve c1 = new Curve();
    int sele = Random(0,2);
    if(sele==0) {
        c1.FromPointSet(pt1,CurveFromPointSetOption.LineFilletComposite,FILLET/5,null,true);
    }
    else {
        c1.FromPointSet(pt1,CurveFromPointSetOption.LineFilletComposite,null,null,true);
    }                       double xA = {};
    double xB = {};
    for(int i=0; i<Hei; i+=4){
        xA[i/4] = ((10-AMP/5)+AMP/5*Cos(FREQOFF+i*(90/50)*FREQ*5))/(i/DECAY*5+1);
    }
    for(int i=0; i<Hei; i+=4){
        xB[i/4] = (-(10-AMP/5)-AMP/5*Cos(FREQOFF+i*(90/50)*FREQ*5))/(i/DECAY*5+1);
    }

    Point pt2 = new Point();
    pt2.ByCartesianCoordinates(csRo,xA,0,Series(0,Hei,4));

    Point pt3 = new Point();
    pt3.ByCartesianCoordinates(csRo,xB,0,Series(0,Hei,4));

    Curve c2 = new Curve();
    c2.FromPointSet(pt2,CurveFromPointSetOption.BSplineCurveByPoints);

    Curve c3 = new Curve();
    c3.FromPointSet(pt3,CurveFromPointSetOption.BSplineCurveByPoints);

    BSplineSurface b1 = new BSplineSurface();
    b1.FromRailsAndSweptSections(c3,c1,c2,null,SkinDirection=DirectionOption.V);

    Solid s1 = new Solid();
    s1.BySurfaceCapping(b1);

    CoordinateSystem csT = new CoordinateSystem();
    csT.ByCartesianCoordinates(cs,0,0,pt2[pt2.Count].Z);
    Solid s2 = new Solid();
    s2.SlabAtCentroid(csT,150,200,pt2[pt2.Count].Z*2);
    s3.BooleanOperation(s2,s1,BooleanOperation.Difference);
}

if(sel==2){
    double x = {};
    double y = {};
```

```
double z = {};
double a = {}; //state

int Nw, Nd, Nh; //number
double W, D, H; //dimension

Nw = Random(4,7);
Nd = Random(4,7);
Nh = Random(5,15);

W = Random(20.0*10.0,50.0*10.0)/10.0;
D = Random(20.0*10.0,50.0*10.0)/10.0;
H = Random(100.0*10,200.0*10)/10.0;

int thresh = 80;

for(int i=0; i<Nw; i++){
    x[i] = {};
    y[i] = {};
    z[i] = {};
    a[i] = {};
    for(int j=0; j<Nd; j++){
        x[i][j] = {};
        y[i][j] = {};
        z[i][j] = {};
        a[i][j] = {};

        for(int k=0; k<Nh; k++){
            x[i][j][k] = W/Nw * i;
            y[i][j][k] = D/Nd * j;
            z[i][j][k] = H/Nh * k;
            a[i][j][k]=0;

            int coreMinW = Nw/2.0-1;
            int coreMaxW = Nw/2.0+1;
            int coreMinD = Nd/2.0-1;
            int coreMaxD = Nd/2.0+1;

            if(i>=coreMinW && i<=coreMaxW && j>=coreMinD && j<=coreMaxD){
                a[i][j][k] = thresh;
            }

            else if(k==0){
                a[i][j][k] = thresh;
            }

            else {
                if(a[i][j][k-1]>thresh){
                    a[i][j][k]=thresh+1;
                }
                else{
                    a[i][j][k] = Random(0,100);
                }
            }
        }
    }
}

for(int i=0; i<Nw; i++){
    for(int j=0; j<Nd; j++){
        for(int k=0; k<Nh; k++){
            if(i==0) a[i+1][j][k]=thresh;
            else if(j==0) a[i][j+1][k]=thresh;
            else if(i==Nw-1) a[i-1][j][k]=thresh;
            else if(j==Nd-1) a[i][j-1][k]=thresh;

            else if(
                a[i][j][k]>=thresh &&
                a[i+1][j][k]<thresh &&
                a[i-1][j][k]<thresh &&
                a[i][j+1][k]<thresh &&
                a[i][j-1][k]<thresh
            )
            {
                int fix = Random(0,4);
                if(fix==0) a[i+1][j][k]=thresh;
                if(fix==1) a[i-1][j][k]=thresh;
                if(fix==2) a[i][j+1][k]=thresh;
```

```
                        if(fix==3) a[i][j-1][k]=thresh;
                    }
                }
            }
        }

        CoordinateSystem csRo = new CoordinateSystem();
        csRo.ByUniversalTransform(cs,0,0,0,0,0,Random(0,180*10)/10);

        CoordinateSystem grid = new CoordinateSystem();
        grid.ByCartesianCoordinates(csRo,x,y,z);

        Solid s = {};
        for(int i=0; i<Nw; i++){
            s[i] = {};
            for(int j=0; j<Nd; j++){
                s[i][j] = {};
                for(int k=0; k<Nh; k++){
                    if(a[i][j][k]<=thresh){
                        s[i][j][k] = new Solid();
                        s[i][j][k].SlabAtOrigin(grid[i][j][k],W/Nw+0.1, D/Nd+0.1, H/Nh);
                    }
                    else{}
                }
            }
        }

        Solid sFlat = new Solid();
        sFlat=Flatten(s);
        Solid sAll = new Solid();
        sAll.SolidUnion(sFlat);
        CoordinateSystem csT = new CoordinateSystem();
        csT.ByCartesianCoordinates(cs,sAll.Centroid.X,sAll.Centroid.Y,H);
        Solid s2 = new Solid();
        s2.SlabAtCentroid(csT,150,200,H*2);
        s3.BooleanOperation(s2,sAll,BooleanOperation.Difference);
    }

    else{}

    BSplineSurface bTemp = new BSplineSurface();
    bTemp.ExtractAllSurfacesFromSolid(s3);
    faceCounter=bTemp.Count;
  }
  return s3;
 };
FunctionArguments = {baseCS};
}
}
```